# NAVAL POSTGRADUATE SCHOOL
## Monterey, California


# THESIS

**A NEW SUFFICIENT CONDITION FOR ROBUST INTERDOMAIN ROUTING**

by

John Henrik Rogers

June 2007

Thesis Advisor: Geoffrey Xie
Second Reader: John Gibson

THIS PAGE INTENTIONALLY LEFT BLANK

# REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188



| 1. AGENCY USE ONLY (*Leave blank*) | 2. REPORT DATE | 3. REPORT TYPE AND DATES COVERED |
|---|---|---|
| | June 2007 | Master's Thesis |

| 4. TITLE AND SUBTITLE A NEW SUFFICIENT CONDITION FOR ROBUST INTERDOMAIN ROUTING | 5. FUNDING NUMBERS |
|---|---|
| 6. AUTHORS Rogers, John | |
| 7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)<br>Naval Postgraduate School<br>Monterey CA 93943-5000 | 8. PERFORMING ORGANIZATION REPORT NUMBER |
| 9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) | 10. SPONSORING/MONITORING AGENCY REPORT NUMBER |

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

| 12a. DISTRIBUTION/AVAILABILITY STATEMENT | 12b. DISTRIBUTION CODE |
|---|---|
| Approved for public release; distribution is unlimited. | |

13. ABSTRACT(*maximum 200 words*)

Border Gateway Protocol (BGP) is currently the only interdomain routing protocol employed on the internet. It allows tens of thousands of Autonomous Systems (ASes) to exchange routing information while implementing economic and organizational policies. However, conflicting policies between ASes can cause routing instability and/or unpredictable routing solutions. A system of routers is robust if routing tables always converge predictably, despite router and link failures. We pursue an approach to guarantee BGP robustness through operational guidelines. Existing guidelines for BGP robustness are essentially geared toward satisfying the same sufficient condition for BGP robustness developed by Griffin and Wilfong. In this thesis, we first show that there exists a weaker sufficient condition for BGP robustness. We then discuss how new guidelines for configuring BGP with a guarantee of robustness may be derived from this new condition. Additionally, we compare various models of BGP behavior and show that the models do not always have equivalent results and sometimes have completely different behavior.

| 14. SUBJECT TERMS Border Gateway Protocol (BGP), interdomain routing, robust routing, stable paths problem, dispute wheel, class-based path-vector system | 15. NUMBER OF PAGES 89 |
|---|---|
| | 16. PRICE CODE |

| 17. SECURITY CLASSIFICATION OF REPORT | 18. SECURITY CLASSIFICATION OF THIS PAGE | 19. SECURITY CLASSIFICATION OF ABSTRACT | 20. LIMITATION OF ABSTRACT |
|---|---|---|---|
| Unclassified | Unclassified | Unclassified | UL |

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18 298-102

THIS PAGE INTENTIONALLY LEFT BLANK

**Approved for public release; distribution is unlimited**

# A NEW SUFFICIENT CONDITION FOR ROBUST INTERDOMAIN ROUTING

John Henrik Rogers
Ensign, United States Navy
S.B., Massachusetts Institute of Technology, 2006

Submitted in partial fulfillment of the
requirements for the degree of

**MASTER OF SCIENCE IN COMPUTER SCIENCE**

from the

**NAVAL POSTGRADUATE SCHOOL**
**June 2007**

Author: John Henrik Rogers

Approved by:

Geoffrey Xie
Thesis Advisor

John Gibson
Second Reader

Peter Denning
Chairman, Department of Computer Science

THIS PAGE INTENTIONALLY LEFT BLANK

# ABSTRACT

Border Gateway Protocol (BGP) is currently the only interdomain routing protocol employed on the internet. It allows tens of thousands of Autonomous Systems (ASes) to exchange routing information while implementing economic and organizational policies. However, conflicting policies between ASes can cause routing instability and/or unpredictable routing solutions. A system of routers is robust if routing tables always converge predictably, despite router and link failures. We pursue an approach to guarantee BGP robustness through operational guidelines. Existing guidelines for BGP robustness are essentially geared toward satisfying the same sufficient condition for BGP robustness developed by Griffin and Wilfong. In this thesis, we first show that there exists a weaker sufficient condition for BGP robustness. We then discuss how new guidelines for configuring BGP with a guarantee of robustness may be derived from this new condition. Additionally, we compare various models of BGP behavior and show that the models do not always have equivalent results and sometimes have completely different behavior.

THIS PAGE INTENTIONALLY LEFT BLANK

# TABLE OF CONTENTS


**I. INTRODUCTION** . . . . . **1**

A. THE IMPORTANCE OF BGP ROBUSTNESS . . . . . 1

B. SUMMARY OF THIS PAPER . . . . . 2

C. ORGANIZATION OF THIS PAPER . . . . . 2

**II. TUTORIAL OF BGP** . . . . . **5**

**III. BACKGROUND WORK** . . . . . **17**

A. APPROACHES TO MAKING EBGP ROBUST . . . . . 17

B. RELATED WORK . . . . . 19

C. THE STABLE PATHS PROBLEM . . . . . 21

D. MODELS OF BGP BEHAVIOR . . . . . 24

1. Simple Path Vector Protocol . . . . . 24

2. Single Node Activation Sequences Model . . . . . 26

3. Multiple Node Activation Sequence Model . . . . . 28

4. Comparison of Models to BGP . . . . . 30

E. ROBUSTNESS . . . . . 30

F. DISPUTE WHEELS . . . . . 30

G. INTERESTING INSTANCES OF SPP . . . . . 32

1. Solvable, but not Safe (SNASM or MNASM or SPVP) . 32

2. Uniquely Solvable, but Not Safe (MNASM) . . . . . 33

3. Categories . . . . . 34

H. HIERARCHICAL BGP . . . . . 35

I. CLASS-BASED PATH-VECTOR SYSTEMS . . . . . 37

**IV. COMPARISON OF BGP MODELS** . . . . . **41**

A. MATCHING PATH ASSIGNMENTS . . . . . 41

B. COMPARISON OF SAFETY . . . . . 46

**V. A WEAKER SUFFICIENT CONDITION FOR ROBUSTNESS 49**
A. MOTIVATION . . . . . 49
B. SUBINSTANCES OF SPP FROM DISPUTE WHEELS . . . . 50
C. ALL DISPUTE WHEELS ROBUST IMPLIES UNIQUELY SOLVABLE . . . . . 50
D. ALL DISPUTE WHEELS ROBUST AND COMPLETE IMPLIES SAFETY . . . . . 53
1. Selecting an Appropriate Model . . . . . 53
2. Complete Dispute Wheels . . . . . 53
3. Existence of Dispute Wheel for an Unsafe Instance of SPP 55
4. All Dispute Wheels Robust and Complete Implies Safety 57
E. A WEAKER SUFFICIENT CONDITION FOR SPP ROBUSTNESS . . . . . 60
F. APPLICATION OF MAIN THEOREM . . . . . 63
1. Finding Dispute Wheels . . . . . 63
2. Constraints that Guarantee Robustness Despite the Presence of a Dispute Wheel . . . . . 63
**VI. CONCLUSION AND FUTURE WORK . . . . . 67**
**APPENDIX. AN EXAMPLE OF A ROUTER CONFIGURATION 69**
**LIST OF REFERENCES . . . . . 71**
**INITIAL DISTRIBUTION LIST . . . . . 73**

# LIST OF FIGURES


1. An Small Scale Example of Internet Routing . . . . . 5
2. Configuration . . . . . 12
3. Available Routes . . . . . 12
4. The Steps in the Permanent Oscillation . . . . . 13
5. A Pictorial Representation of SPP . . . . . 23
6. SPVP Process at Node $u$ from [Ref. 12] ) . . . . . 25
7. The SNASM Routing Protocol Dynamics . . . . . 28
8. The MNASM Routing Protocol Dynamics . . . . . 29
9. A Generalized Dispute Wheel . . . . . 31
10. SPP Instance NEXT . . . . . 32
11. An Instance of SPP that is Uniquely Solvable, but Not Safe (Naughty Gadget from [Ref. 12] ) . . . . . 33
12. Properties of SPP Instances . . . . . 35
13. An Instance of SPP That Shows MNASM Does Not Match SPVP . . . 41
14. A Sequence of Path Assignments Given by SPVP for Figure 13 . . . . 43
15. How Models Match Each Other over the Space of Instances of SPP and Initial Path Assignments . . . . . 46
16. Instance of SPP for Theorem IV.10. DISAGREE from [Ref. 12] . . . . 48
17. Safety Between Different Models . . . . . 48
18. An Instance of SPP that has a Dispute Wheel, but is Robust . . . . . 49
19. The Dispute Wheel of Figure 18 . . . . . 50
20. Illustration for Theorem V.1. The nodes inside the dashed circle all represent nodes belonging to $T$. Outside the circle are edges and nodes in $T_1$ and $T_2$. Dashed Edges are in $T_2$. Solid edges are in $T_1$ . . . . . 51
21. An Illustration of Some Terminology . . . . . 54

22. COUNTEREX: A Robust Instance of SPP that Does Not Meet Our Condition . . . . . 62
23. Conditions to Guarantee SPP Robustness . . . . . 62
24. Robust Operational Guidelines for SPP and BGP . . . . . 66

# LIST OF TABLES

I. Unsafe Path Assignments for NEXT . . . . . 33

II. Path Assignments for Naughty Gadget . . . . . 34

THIS PAGE INTENTIONALLY LEFT BLANK

# ACKNOWLEDGMENTS

I would like to thank Geoffrey Xie for all the wonderful discussions we had during our work together. He was always able to clear up areas that I misunderstood, provide insight into new areas, and keep me motivated.

I would like to thank my loving parents, Kristina C. Rogers and CAPT T. Wayne Rogers (NPS '72). They've maintained a constant interest in my education for more than two decades.

THIS PAGE INTENTIONALLY LEFT BLANK

# I. INTRODUCTION

Border Gateway Protocol (BGP) is currently the only interdomain routing protocol employed on the internet. It allows hundreds of thousands of autonomous systems (ASes) to interconnect by providing a common protocol to share network reachability information. Within an autonomous system, shortest path routing protocols are sensible. They provide a predictable method of routing network traffic and usually provide optimal routing. However, using shortest path heuristics to route traffic between autonomous systems is unattractive. Each AS is administered by an organizational entity that may have a range of economic and organizational incentives. Because the goal of many ASes is to earn income by providing internet service, these incentives vary widely between ASes. Furthermore, even an AS which does not seek to gain financially, may wish to limit unnecessary network traffic flow so that it maintains an acceptable level of service to its users. The incentives of ASes can be expressed in terms of network policies. The varying policies between ASes create the need for a protocol which does not rely on shortest path routing.

BGP has been widely successful because it gives network administrators the ability to interconnect with other ASes and implement their organization's policies. Unfortunately, the ability of BGP to implement organizational policies may also lead to routing oscillations and unpredictable routing solutions [Ref. 24] when ASes have conflicting policies. We describe a system of routers as *robust* if routing tables always converge predictably, under any set of router and link failures.

## A. THE IMPORTANCE OF BGP ROBUSTNESS

Robustness is crucial for the performance of the internet infrastructure. Persistent routing oscillations may significantly impact end-to-end performance, resulting in increased latency and dropped packets. Persistent routing oscillations also make it difficult for network operators to identify, debug, and correct undesirable rout-

ing instances. Furthermore, robustness is crucial for maintaining predictable routing behavior. If routing behavior is unpredictable, optimal routing may not be achieved.

## B. SUMMARY OF THIS PAPER

A number of approaches have been pursued to address BGP instability. This paper investigates achieving robustness of eBGP sessions by implementing local and global constraints. Using the stable paths problem as a framework for BGP polices [Ref. 12], we investigate and compare various BGP models to show that they do not always match each other. We present new sufficient condition for robustness, that is weaker than any previously published condition. We pursue devising constraints which guarantee this condition. We also apply our results using the class-based path-vector system [Ref. 18].

## C. ORGANIZATION OF THIS PAPER

The remainder of this paper is organized as follows.

Chapter II gives a tutorial of BGP. We introduce BGP and the services that it provides. We describe how routers establish BGP sessions and describe the various messages that can be exchanged. We discuss how BGP allows operators to implement network policy. We discuss how routers use BGP to store, select, and advertise routes. We define three major design goals of BGP: autonomy, expressiveness, and robustness. We detail how permanent routing oscillations may arise from conflicting policies. We discuss route flap dampening as the current solution to address BGP oscillations.

Chapter III presents background work that addresses achieving BGP robustness. We review the main approaches to making BGP robust. We discuss why we pursue an approach to achieving BGP robustness that relies on operational guidelines and global constraints. We give a summary of related work on BGP. We reintroduce the stable paths problem as a framework to model policies and routing solutions of BGP systems. We define solvability as the existence of a stable routing assignment.

We describe three models of BGP behavior: the simple path vector protocol, the single node activation sequence model, and the multiple node activation sequence model. For each model we define safety. We reintroduce the dispute wheel as a sufficient condition for the robustness of the stable paths problem. A dispute wheel represents a set of mutually conflicting policies. We discuss the hierarchical BGP model which describes local and global constraints on ASes to guarantee robustness. We reintroduce the class-based path-vector system which describes generalized local and global constraints on ASes that guarantee robustness.

Chapter IV compares the three models of BGP behavior. We describe how the models match each other in terms of achieving similar successive path assignments when given the same instance of the stable paths problem and initial routing tables. We prove that while some models match each other, others do not. We compare the definitions of safety between the different models. We prove that while the definition of safety in one model may imply safety in another, this is not true for all models.

Chapter V gives our main result. We motivate our result by an instance of the stable paths problem which is robust, but contains a dispute wheel. We introduce a new condition on instances of the stable paths problem. We prove that this condition is robust, and weaker than previously published conditions. We investigate applying our result using the class-based path-vector system framework. We pursue devising broader guidelines to guarantee robustness, despite the presence of a dispute wheel.

Chapter VI gives conclusions and future work.

THIS PAGE INTENTIONALLY LEFT BLANK

# II. TUTORIAL OF BGP

Border Gateway Protocol (BGP) is currently the only interdomain routing protocol employed on the internet. The internet connects tens of thousands of autonomous systems (ASes). An AS is a collection of routers controlled by a single entity, such as a local ISP or university. It is also common for very large organizations to operate more than one AS. Each AS is given a globally unique number called the autonomous system number (ASN). Inside an AS an interior gateway protocol (IGP) such as RIP and OSPF is used to determine routes. However, ASes communicate with each other using BGP, making BGP an interdomain protocol. Specifically, BGP gives each AS the ability to (1) obtain reachability information from neighboring ASes (2) propagate routing information and (3) choose routes based on reachability and policy [Ref. 22]. Unlike OSPF or RIP, routes in BGP are not usually determined by shortest path metrics. ASes often have various economic incentives. Because BGP gives network administrators an enormous amount of control over how routes are advertised to neighboring ASes and how routes are chosen, BGP is often referred to as "policy-based" routing.

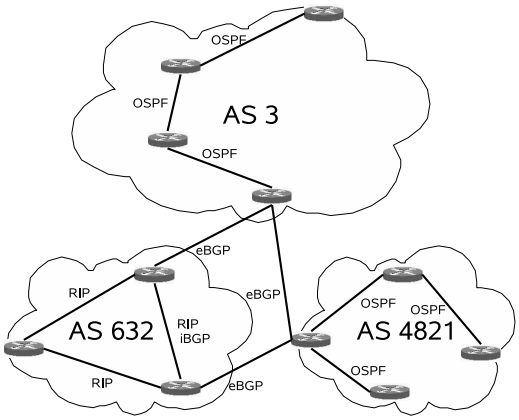


Figure 1. An Small Scale Example of Internet Routing

To begin a BGP session, a BGP speaker establishes a TCP connection on

port 179 with another BGP speaker and sends an OPEN message. There are two types of BGP sessions. An interior border gateway protocol (iBGP) session allows an autonomous system to propagate routing information within itself. An exterior border gateway protocol (eBGP) session allows an autonomous system to share routing information with a different AS, also known as an external peer. See Figure 1 for a small scale example of routing protocols used on the internet. For the purposes of this paper, we will ignore the complexities of iBGP and assume that each AS has completely uniform routing information at any given time. Therefore, we consider each AS as a single entity or node that has eBGP sessions with external peers.

Once a BGP session has been established, several types of messages are sent between BGP speakers. KEEPALIVE messages are periodically to ensure that the connection is alive. NOTIFICATION messages are sent in response to errors or special conditions. UPDATE messages are used to advertise routes between BGP speakers. A route is a set of destinations with information about the path to those destinations. UPDATE messages send information about routes by using the Network Layer Reachability Information (NLRI) field and the path attributes field[Ref. 22].

The path attributes field of an UPDATE message allows BGP speakers to share detailed information about routes. We will briefly discuss some of the most important attribute types, including AS_PATH, ORIGIN, MULTI_EXIT_DISC, and LOCAL_PREF. The mandatory AS_PATH attribute informs the local BGP speaker of which ASes carried the routing information to the local speaker. If this routing information has not changed, these same ASes will carry any traffic sent to the route's destination. The ability to share the AS_PATH parameter makes BGP a path-vector protocol. When an AS shares reachability information about a destination to one of its neighbors, it shares the entire path of ASes to the destination. This helps prevent routing loops, because no path will ever be accepted if it crosses through the same AS number twice. The mandatory ORIGIN attribute identifies whether the original source of routing information was from an interior gateway protocol, the exterior

gateway protocol, or unknown. The optional MULTI_EXIT_DISC (MED) attribute is passed between external peers and allows a local AS to discriminate between multiple entry and exit points to the same neighboring AS. The LOCAL_PREF attribute must be included in any UPDATE message between internal peers. This attribute helps an AS rank paths and maintain consistent rankings throughout the AS.

As discussed above, BGP is policy-based routing. BGP operators use rankings and filters to implement their policies. A BGP speaker may have a multiple routes to a single destination available. Rankings determine which of these routes should be used. Also, an AS may not want to share all of its routes with an external peer. Export filters allow an AS to place controls on the routes advertised to external peers. Conversely, an AS may not want to use some of the routes that it has received. Import filters allow an AS to not use specified routes.

Rankings are determined from a large number or factors. Phase 1 of the decision process is decision function that is invoked whenever a BGP speaker recieves an UPDATE message, from a peer, that advertises a new route, a relacement route, or a withdrawn route. Phase 1 calculates the degree of preference for each newly recieved or replaced route. If the route is learned via an iBGP session, either the LOCAL_REF attribute is is taken as the degree of preference or the degree of preference is computed based on preconfigured policy information. If the route is learned via an eBGP session, then the degree of preference is based on preconfigured policy information. [Ref. 22].

Phase 2 of the decision process is invoked immediately after Phase 1 and determines which routes should be used by a BGP speaker. AS loops are detected by scanning the full AS path of each route and making sure that none of these ASNs matches that of the local system. [Ref. 22]. Also, if a route becomes inaccessible, it can not be used. Once these routes have been eliminated, the highest ranked route is selected by the following rules in their exact order [Ref. 22]:

1. Prefer the path with the largest local preference.

2. Remove from consideration all paths that are not tied for having the largest local preference.
3. Prefer the path that passes through the smallest number of ASes.
4. Remove from consideration all paths that are not tied for passing through the smallest number of ASes.
5. Prefer the path that has the lowest Origin number.
6. Remove from consideration all paths that are not tied for having the lowest Origin number.
7. Prefer the path with the lowest MED attribute.
8. Remove from consideration all paths that are not tied for having the lowest MED attribute.
9. If at least one path was received via EGP, remove from consideration all paths that were received via IGP.
10. Prefer the route with the most preferred interior cost.
11. Remove from consideration all paths that are not tied from having the the most preferred interior cost.
12. Prefer the route with the lowest BGP identifier value.
13. Prefer the path with the lowest external peer IP address.

Within an AS, BGP speakers may assign routes a specific local preference value, based on criteria such as AS_PATH. Because local preference is the first attribute inspected in the decision process, this ability allows every AS to rank all routes in any arbitrary order.

A BGP speaker may be configured to filter routes in a number of ways. Filters may be specified by ASNs occurring in a route's the AS_PATH attribute and/or the route's destination address. Filters may be applied to prevent a route from entering the router's routing information base. This would prevent a specified route from ever being selected. Filters may also be applied to prevent a route from being sent in an UPDATE message to an external peer.

Now that we have discussed ranking and filtering, we discuss a conceptual model of how BGP stores, selects, and advertises routes. There are three conceptually distinct storage tables for routes. The Adj-RIBs-In table contains all unprocessed routes that have been received from peers. The Loc-RIB table contains each actual route used locally for all available destinations. This is determined by applying import filters and rankings. The Adj-RIBs-Out contains the routing information that will be shared with neighbors in outgoing UPDATE messages. Suppose a BGP speaker receives a route from a peer in an UPDATE message. The BGP speaker will store the route in the Adj-RIBs-In table. Next, the BGP speaker will undergo its decision process to determine if this received route should be used. Routes which should be filtered and routes which have a repeated ASN are eliminated from the decision process. The router will use its ranking rules to determine whether the received route is now the highest ranked route to a destination. If this is the case, the received will replace the existing route in the Loc-RIB table and the BGP speaker will begin routing traffic towards the first hop of the new route. Finally, export filters are applied to determine whether the route should be advertised to neighbors. If the route is eligible to be advertised to neighbors, the route will be updated with new attributes such as AS_PATH and NEXT_HOP. The updated route and eligible neighbors will be stored in the Adj-RIBS-Out table. UPDATE messages will be sent containing the updated route.

Routes may also be withdrawn in three different ways. If a route is withdrawn, the route must be deleted from Adj-RIBS-In, Loc-RIB, and Adj-RIBS-Out. If a BGP speaker deletes any route from the Adj-RIBS-Out table, it must inform its neighbors that this route is no longer available. A route can be withdrawn by sending an UPDATE message with the route placed in the WITHDRAWN ROUTES field. A route can be withdrawn by advertising a new route that contains the same NLRI. A route can be withdrawn by closing the BGP connection.

The ability of BGP to function as a policy-based protocol leads us to introduce

two major design goals of BGP, **autonomy** and **expressiveness**. Autonomy is the ability of network operators to make policy decisions without coordinating with other ASes. Without a large amount of autonomy, network operators may have to update their policies when the BGP configurations of neighboring ASes change. Furthermore, without a large amount of autonomy, network administrators of different ASes may be forced into a situation where they must disclose some of their policies to each other. Due to economic incentives, network operators often require that they keep their BGP policies private. Expressiveness is the ability of network operators to specify network policy in a flexible manner. For instance, shortest-path routing does not provide enough expressiveness, because it can't capture the economic relationships between many ASes such as customer, provider, and peer [Ref. 5] [Ref. 11].

In general, BGP operators configure policies in line with their organization's economic incentives, which are determined by agreements with neighboring ASes. Many agreements between ASes can be characterized as either a peer-to-peer relationship or a customer-provider relationship [Ref. 17]. In a peer-to-peer relationship, two neighboring ASes benefit from exchanging traffic between each other's customers. When BGP relationships are discussed, the word "peer" will refer to an AS which is following a peer-to-peer agreement with a neighboring AS. In a customer-provider relationship, one neighbor takes on the role of customer and the other takes on the role of provider. The customer pays the provider for access to internet destinations that could not be otherwise obtained [Ref. 8]. If an organization has such agreements, network operators may implement an economically advantageous policy by adhering to the following rules:

1. An AS can advertise only the routes of itself and its customers to a provider or peer.
2. An AS can advertise all known routes to its customers.

The first rule prevents an AS from carrying traffic without receiving compensation or benefit. The second rule allows a provider to inform its customers of routes

so that it may receive compensation for carrying traffic.

Routing Oscillations occur when routers exchange streams of routing updates that do not reflect any change to network topology or configuration. Some oscillations, such as the RIP v1 count to infinity problem, eventually end after a large amount of unnecessary information has been exchanged. An oscillation that will eventually end is known as a transient oscillation. Permanent oscillations occur when routers exchange endless streams of routing updates, and may be created by conflicting BGP policies or iBGP configurations. Routing oscillations may use up router processing power, increase network latency, cause forwarding loops and partition the network [Ref. 25]. Furthermore, oscillations can be exacerbated by failed links as well as complicate the diagnosis and debugging of network problems [Ref. 23]. Finally, routing oscillations may significantly affect the increasing number of streaming media applications on the internet today.

Some BGP oscillations may arise from iBGP configurations alone. Clustering-induced divergence occurs when an interaction between route reflection clustering and intradomain routing costs causes permanent oscillations [Ref. 15]. This anomaly may occur even when eBGP configuration is robust. Griffin *et al* [Ref. 15] gave a sufficient condition to solve this problem, which is based upon restricting the choices of paths at some routers. In another type of iBGP anomaly, MED-induced divergence occurs when an interaction between MED values, route reflection clustering, and intradomain routing costs causes permanent oscillations [Ref. 2]. Musunuri and Cobb proposed routing protocols that would eliminate this anomaly [Ref. 20] .

Besides oscillations occurring from iBGP, eBGP may also cause oscillations. When multiple BGP speakers have conflicting routing policies, there may be permanent oscillations. To see how router configuration may lead to permanent routing oscillations, consider a case where there are four eBGP speakers named “0”, “1”, “2”, and “3” with the unique ASNs 100, 101, 102, and 103 respectively. The system has the configuration as depicted in Figure 2. We are interested in routing a particular

packet to a destination d inside ASN 0. Therefore, we are interested in routes that have an AS Path that ends in 100. With that in mind, each router is configured to have the routes and preferences as depicted in Figure 3.

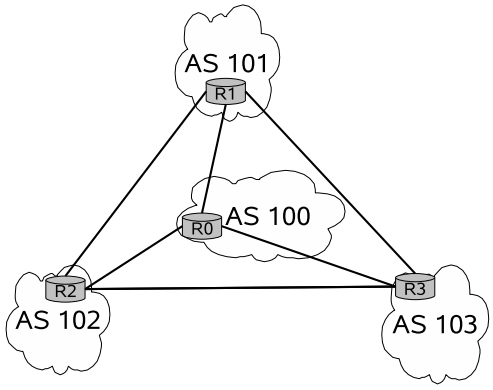


Figure 2. Configuration

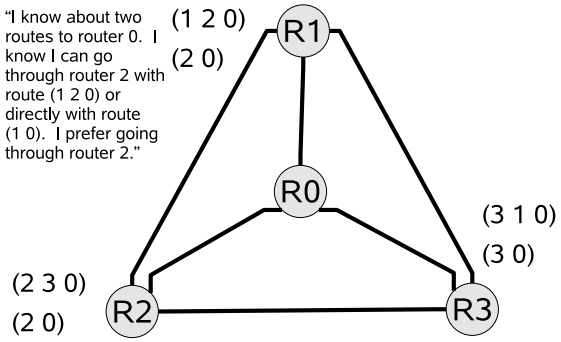


Figure 3. Available Routes

The router configurations can be described as follows. Router 0 exports all routes to routers 1, 2, and 3. Router 1 exports all routes to routers 0, 1, and 2. Router 1 filters all routes received from router 3 and filters the single route received from router 2 that has the AS Path 101 102 103 100. Router 1 prefers the route with the AS Path 101 102 100 to the route with the AS Path 101 100. Router 2 exports all routes to routers 0, 1, and 3. Router 2 filters all routes received from router 1 and

filters the single route received from router 3 that has the AS Path 102 103 101 100. Router 2 prefers the route with AS Path 102 103 100 to the route with AS Path 102 100. Router 3 exports all routes to routers 0, 1, and 2. Router 3 filters all routes received from router 2 and filters the single route received from router 1 with AS Path 103 101 102 100. Router 3 prefers the route with AS Path 103 101 100 to the route with AS path 103 100. For an example configuration file for router 1, see the appendix.

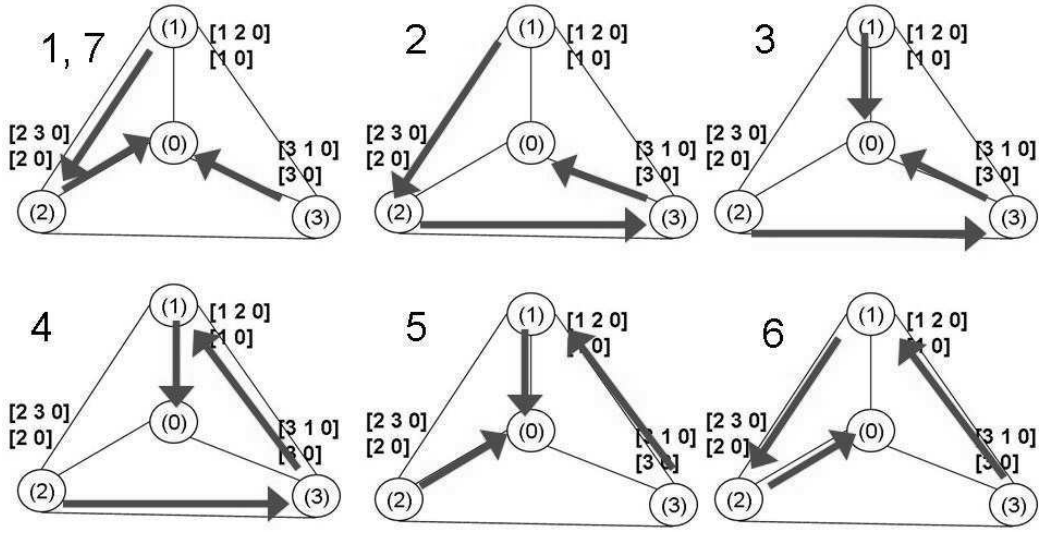


Figure 4. The Steps in the Permanent Oscillation

The router configuration discussed above will always give rise to permanent routing oscillations. We will go through a sequence of routing updates to show the permanent oscillations as depicted in Figure 4.

1. Router 1 routes through router 2 to router 0. Router 2 routes directly to router 0. Router 3 routes directly to router 0. Also, an UPDATE message has been sent from router 3 to router 2, informing router 2 of its new route to router 0. However, router 2 has not received this message yet.

2. Router 2 receives and processes the UPDATE message from router 3. Router 2 changes its route to route through router 3 to router 0. Router 2 sends an UPDATE message to router 1 informing router 1 of its new route. However, router 1 has not processed this message yet.

3. Router 1 receives and processes the UPDATE message from router 2. Router 1's current route to router 0 through router 2 is no longer available and the new route received from router 2 is filtered. Therefore router 1 changes its route and routes directly to router 0. Router 1 sends an UPDATE message to router 3 informing router 3 of its new route.

4. Router 3 receives and processes the UPDATE message from router 1. Router 3 changes its route and routes through router 1. Router 3 sends and UPDATE message to router 2 informing router 2 of its new route.

5. Router 2 receives and processes the UPDATE message from router 3. Router 2's current route to router 0 is no longer available. Router 2 changes its route to route directly to router 0. Router 2 sends an UPDATE message to router 1 informing router 1 of its new route.

6. Router 1 receives and processes the update message from router 2. Router 1 changes its route and routes through router 2. Router 1 sends an UPDATE message to router 3 informing router 3 of its new route.

7. Router 3 receives and processes the update message from router 1. Router 3's current route is no longer available. Router 3 changes its route to route directly to router 0. Router 3 sends an UPDATE message to router 2 informing router 2 of its new route. Note that at the end of step 7 we are in the exact same state as in the end of step 1.

Because Step 1 and Step 7 result in the exact same state, this process will repeat itself indefinitely. If a system of routers is always guaranteed to converge and stop changing routes, no matter what order messages are processed in, the system is known as **safe**. The example we have just examined is not safe. Solvability is another characteristic of systems of BGP routers that does not always hold. A system of routers is **solvable** if there exists a set of system wide routing tables where if any router receives a correct UPDATE message, that router will not change its current routing table. The example we have just examined is not solvable. **Unique solvability** is a more stringent characteristic where there is exactly one set of system-wide routing tables that are solvable. If a system of routers is uniquely solvable and safe, the system is guaranteed to converge in a predictable manner.

Now that BGP oscillations have been discussed, we introduce **robustness** as the third major design goal of BGP. Robustness is a characteristic where router con-

figuration can not lead to routing oscillations and must always produce a predictable, unique routing solution, under any set of link and router failures. For BGP to be robust, constraints must be put on the expressiveness and autonomy of the protocol [Ref. 11].

In order to minimize the effects of BGP oscillations, route flap dampening [Ref. 25] is often employed. Route flap dampening is an extension to BGP that allows routers to maintain information on the stability of individual routes. A BGP speaker will suppress routes that show a large degree of instability. Also, fixed timers may be used to slow route advertisement. While route flap dampening may successfully minimize some of the adverse effects of oscillations, it does not provide a complete solution. Route flap dampening causes oscillations to run in slow motion and does not guarantee that routing tables will converge to a predictable, unique solution.

In this chapter, we have examined how BGP uses rankings and filterings to select routes and implement network policy. We have described how conflicts in policies may create BGP oscillations.

THIS PAGE INTENTIONALLY LEFT BLANK

# III. BACKGROUND WORK

## A. APPROACHES TO MAKING EBGP ROBUST

There are currently three main approaches to address the instability of eBGP [Ref. 12]. The approaches consist of operational guidelines for BGP operators, static analysis of routing policies, and modification of the BGP protocol.

If every BGP operator followed the same set of operational guidelines, it is possible to prove the robustness of BGP for certain sets of guidelines. For instance, if every BGP operator configured policies using route filtering alone, BGP is guaranteed to be a robust protocol [Ref. 12]. Another flexible, but complex set of robust guidelines are proposed by Gao and Rexford [Ref. 8].

There are a number of downsides to relying on operational guidelines. First, not all BGP operators may follow such operational guidelines. A set of operational guidelines may not capture every policy that BGP operators may be interested in implementing or BGP operators may ignore operational guidelines altogether. Second, the set of robust operational guidelines may be overly strict. There may exist configurations of routers that are robust, despite the fact they do not implement any known operational guidelines. Third, operational guidelines may require BGP operators to disclose some amount of policy with each other in order to check global constraints. For reasons already noted, most BGP operators are very reluctant to disclose their configurations with each other.

In another approach, BGP robustness could be achieved by static analysis of router configurations. Such a solution would analyze the configuration of all BGP speakers and look for policy conflicts. This solution has been proposed by Govindan *et al.* [Ref. 10]. There are at least two major drawbacks to this approach. First, BGP operators would have to disclose the policies and configurations of their AS with each other. For economical reasons, most BGP operators are very reluctant to disclose their configurations. Second, such an approach is likely to be intractable, without any

heuristic procedure to check convergence properties or constraints on ASes. Griffin and Wilfong have shown that checking for global convergence conditions is either NP-complete or NP-hard [Ref. 16].

In the final approach, the BGP protocol could be modified to suppress or prevent eBGP oscillations. This approach is sometimes referred to as a *dynamic* approach, because it happens at run-time. As discussed in Chapter II, the route-flap dampening [Ref. 25] can suppress eBGP oscillations. Unfortunately, this approach only makes oscillations run in slow motion and does not guarantee that BGP will converge to a predictable, unique solution.

More extensive modifications to BGP have also been proposed. Griffin and Wilfong propose a modification to BGP where an attribute called *path history* is used to identify paths whose histories contain cycles. This attribute is exchanged between BGP speakers. Once these paths have been identified, the modified protocol can also suppress such paths [Ref. 14]. Another somewhat similar modification to BGP has been proposed by Tien Ee *et al.* [Ref. 4]. They proposed a mechanism whereby route advertisements are tagged by a global precedence value. When a BGP speaker advertises this route to its neighbors, it will increment this value by a number corresponding to its LOCAL_PREF for that route. If permanent BGP oscillations occur, routers will rely on these global precedence values instead of the local degree of preference, creating a stable path assignment.

There are several drawbacks to solutions which modify BGP. First, in the two previous BGP modifications discussed, *every* BGP speaker must implement the proposed protocol to prevent BGP oscillations. There are hundreds of thousands of BGP speakers deployed on the internet today and operators may be unwilling to update their routers to new standards. Second, protocol modifications that suppress routes dynamically are unpredictable by nature. Often, it is impossible to predict exactly which BGP speaker will begin suppressing routes related to permanent oscillations, eliminating the possibility of robustness. Finally, protocol modifications that

suppress routes involved in conflicting policies often sacrifice a large degree of transparency [Ref. 11]. Transparency is the ability of BGP operators to to understand how the policies they have written affect the routing protocol and routing tables. When dynamic solutions suppress routes, it becomes difficult for BGP operators to maintain and debug routing policies.

Based upon the above discussion, we pursue an approach that relies on operational guidelines along with global constraints in order to achieve robustness.

## B. RELATED WORK

Bertsekas [Ref. 1] proved that the distributed Bellman-Ford algorithm converges. Because BGP has the ability to employ policy based routing, this proof of shortest path routing does not apply to BGP in general.

Varadhan *et al.* [Ref. 24] first observed that conflicting policies in BGP configuration could lead to persistent routing oscillations. Furthermore, they introduced the concept of safety, by defining an AS as "safe" if the policy of an AS does not cause oscillations. They also speculated that only shortest path route selection is provably safe.

Labovitz *et al.* [Ref. 19] presented results from a two year long study of internet routing convergence. They discussed the theoretical upperbound of convergence time for certain systems. They showed that when routing faults were injected into the internet, convergence took much longer than previously thought.

Feigenbaum, Sami, and Shenker [Ref. 6] showed that systems with next hop rankings always have at least one stable routing. However, because of the distributed nature of BGP, such systems are not guaranteed to converge to a stable routing. We give an example such a system in Figure 10.

Gao and Rexford [Ref. 8] introduced sufficient conditions on topology, filtering, and rankings to guarantee routing stability and safety. These conditions reflect the real-world configuration of autonomous systems. They introduced and defined

the activation sequence in order to model the behavior of BGP. They developed a system of constraints based upon the principle that every autonomous system should regard each of its neighbors as either a provider, a customer, or a peer. Furthermore, they defined a series of constraints based on each of these relationships. Finally, they proved that if every AS follows these constraints, stable internet routing can be achieved without global coordination. Unfortunately, ASes do not always follow such guidelines. Further work by Gao [Ref. 7] showed that some small ISPs do not follow the guidelines.

Griffin, Shepherd, and Wilfong [Ref. 12] introduced the dispute wheel as a sufficient condition for robustness. They defined the stable paths problem (SPP), which is discussed in more detail in this chapter. They also used the simple path vector protocol (SPVP) [Ref. 9] to model the behavior of BGP. They showed that determining the solvability of SPP is an NP-complete problem. Furthermore, they introduced the dispute wheel. They proved that the absence of a dispute wheel is a sufficient condition for SPP solvability, safety, and robustness.

Griffin, Jaggard, and Ramachandran [Ref. 11] introduced a framework to describe class-based path-vector systems. They detailed a method where matrices are used to describe the scoping (also known as filtering) and ranking rules of an AS based upon its relationships with neighboring ASes and hierarchical level. They showed how the framework could be used to describe conditions on relationships like those proposed by Gao and Rexford [Ref. 8]. They also discussed the design goals for path-vector protocols like BGP. They showed that in order to guarantee robustness, there is an inherent tradeoff between expressiveness and the need for global constraints. They showed that if full autonomy was allowed in a system, autonomous systems could only express rankings based on shortest paths.

Jaggard and Ramachandran [Ref. 18] continued work on class-based path-vector systems by giving specific global constraints on a system that guarantee robustness. They proved an exact global condition for the creation of a dispute wheel.

Furthermore, they gave polynomial-time central and distributed algorithms to enforce this constraint. Unfortunately, their constraint is not likely to be the most general constraint for path-vector systems.

Feamster, Johari, and Balakrishnan [Ref. 5] explored the inherent tradeoff between autonomy and expressiveness. They showed that next-hop rankings were not safe.

## C. THE STABLE PATHS PROBLEM

The Stable Paths Problem (SPP) captures the apparent routing policies over a network of autonomous systems running BGP [Ref. 12].

The SPP framework is designed to describe the most important features of path selection in BGP. The SPP framework consists of a simple, bidirectional graph $G$, which contains a collection of vertices $V$ and edges $E$. There is a vertex denoted 0 which represents the origin. Every other vertex is interested in finding a path to the origin. For each vertex $v \in V$, $\mathcal{P}^v$ represents a set of paths that are available from that vertex $v$ to the origin 0.

The SPP framework also includes $\Lambda$ , which is a ranking function on the paths $\mathcal{P}^v$ available at each vertex $v \in V - \{0\}$. Let $\mathcal{P}$ be the set of all paths available at all vertices. Because the set of routes $\mathcal{P}^u$ available at each vertex $u$ may be limited, SPP captures the ability of each AS to filter routes. However, the SPP framework does not specify whether a route has been filtered by an import filter or an export filter. For each node $v$, there is a ranking function $\lambda^v$, that is defined over $\mathcal{P}^v$. Let $\Lambda = \{\lambda^v | v \in V - \{0\}\}$. For each such node $v$ , if $P_1, P_2 \in \mathcal{P}^v$ and $\lambda^v(P_1) > \lambda^v(P_2)$ then node $u$ is said to prefer the path $P_1$ over the path $P_2$. The ranking function $\Lambda$ captures the ability of each AS to autonomously and expressively rank routes. Formally, an instance of the stable paths problem denoted $S$ is expressed as a triple $S = (G, \mathcal{P}$ $, \Lambda)$.

Finally, we make several assumptions about the paths permitted at every node and the ranking function. We assume that $\mathcal{P}^0 = \{(0)\}$. For all $u \in V - \{0\}$ we assume:

1. If a path is permitted $P \in \mathcal{P}^u$, then $P$ is a simple path. (simplicity, no repeated nodes)
2. $\epsilon \in \mathcal{P}^u$ (empty path permitted)
3. $\Lambda^u(\epsilon) = 0$ and $\forall P \in \mathcal{P}^u$ such that $P \neq \epsilon$, $\Lambda^u(P) > 0$(empty path lowest ranked)
4. If $P_1$ , $P_2 \in \mathcal{P}^u$, $P_1 \neq P_2$ , and $\lambda^u(P_1) = \lambda^u(P_2)$, then $\exists w \in V$ such that $P_1 = (uw)P_1^l$ and $P_2 = (uw)P_2^l$ where $P_1^l$ and $P_2^l$ are subpaths of $P_1$ and $P_2$ respectively. (strictness, two identically ranked paths have the same next hop)

Rule 1 captures the fact that BGP eliminates paths with repeated AS numbers. Rule 2 captures the fact that it is possible for every AS to not be able to reach any arbitrary destination. Rule 3 captures the fact that an AS will take any allowed and available path to a destination rather than leave the destination unreachable. Rule 4 captures the fact that when an AS receives routes from two different ASes, one route must be preferred over another.

Figure 5 gives a pictorial representation of the SPP. In this figure we see that the vertices $V$ consist of $\{0, 1, 2, 3\}$ and the edges $E$ consist of $\{(10)(12)(13)(20)(23)(30)\}$. At vertex 1, the paths to the origin (10) and (120) are available. Vertex 1 would prefer to reach the origin through vertex 2 by using path (120) rather than reach the origin directly using path (10).

A path assignment $\pi$ is a function that maps each node $u \in V$ to a path $\pi(u) \in \mathcal{P}^u$. The set of paths, choices$(\pi, \mathrm{u})$, is defined to be

$$\text{choices}(\pi, \mathrm{u}) = \begin{cases} \{(uv)\pi(v) | \{u, v\} \in E\} \cap \mathcal{P}^u & u \neq 0 \\ \{(0)\} & u = 0 \end{cases}$$

Note that, only the path of length 1, (0), is allowed at the origin.

Suppose $\mathcal{R}^u \subset \mathcal{P}^u$ such that each path has a distinct next-hop. The *best path in* $\mathcal{R}^u$ is defined to be

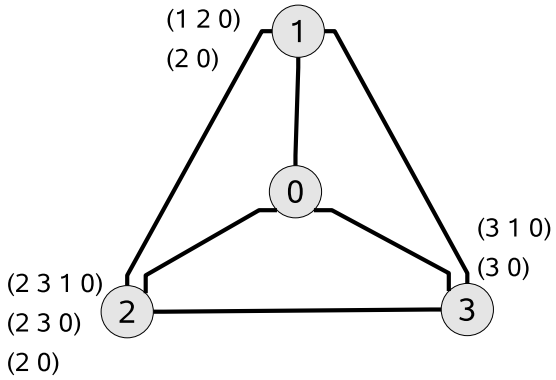


Figure 5. A Pictorial Representation of SPP

$$\text{best}(\mathcal{R}^u) = \begin{cases} P \in \mathcal{R}^u \text{ with maximal } \lambda^u(P) & \mathcal{R}^u \neq \emptyset \\ \epsilon & \mathcal{R}^u = \emptyset \end{cases}$$

The path assignment $\pi$ is *stable at node n* if $\pi(u) = \text{best}(\text{choices}(\pi, \text{u}))$. The path assignment $\pi$ is stable if it is stable at each node $u \in V$. As mentioned in [Ref. 12] any stable path assignment also describes a tree containing the origin.

An instance of SPP is *solvable* if there exists a stable path assignment for the instance. An instance of SPP is *uniquely solvable* if there exists exactly one stable path assignment of the instance.

Deriving subinstances of SPP will be used in later sections represent an instance of the stable paths problem where nodes or links have failed. Given an instance of SPP $S = (G, \mathcal{P}, \Lambda)$, where $G = (V, E)$, there is a natural way to derive subinstances of SPP from subsets of $E$. Suppose $E' \subset E$, we define $\text{SPP}(E') = (G_{E'}, \mathcal{P}_{E'}, \Lambda_{E'})$ to be the derived instance of SPP from $E'$. Let the graph be $G_{E'} = (V, E')$ . Let the set of available paths $P_{E'} = \{P | P \in \mathcal{P}\}$ and every edge in the path $P$ is present in $E'$ . For each node $u$, we will denote its set of available paths as $\mathcal{P}^u_{E'}$. Finally, let the ranking function be $\Lambda_{E'} = \Lambda$, but modified to exclude all omitted paths.

## D. MODELS OF BGP BEHAVIOR

There are three models of BGP behavior, the simple path vector protocol, the single node activation sequences model, and the multiple node activation sequences model. All three models can be expressed in terms of the stable paths problem and can express how BGP speakers exchange UPDATE messages and update their routing tables. Furthermore, all models have specific definitions for safety, which are all conceptually equivalent to BGP safety. In Chapter IV, we will investigate the equivalence of the models.

### 1. Simple Path Vector Protocol

The simple path vector protocol (SPVP) captures the most important behavioral characteristics of BGP [Ref. 13] [Ref. 16] . It is a distributed algorithm which tries to solve the stable paths problem. The protocol will always diverge if an instance of SPP is not solvable. However, as we will see later, the protocol can also diverge for an instance of the stable paths problem that is solvable.

It will be necessary to reintroduce much of the notation from [Ref. 12]. Each node $u$ can store information about paths in two different data structures. The data structure rib(u) stores $u$'s current path to the origin or $\pi(u)$. For each node $u$ and a stable paths problem $S$, we define the set of nodes peers(u) to be the set $\{v|(u\ v) \in E\}$. For each $w \in$ peers(u), the data structure rib-in($u \Leftarrow w$) stores the most recently received and processed path from $w$. Because we do not assume messages are processed immediately, it is possible that rib-in($u \Leftarrow w$) might contain a different, older path than rib(w). Therefore, we define the choices of paths available for a node running SPVP slightly differently than we do for the stable paths problem in general. Under SPVP, we define the path choices available at node $u$ to be:

$$\text{SPVP-choices}(u) = \{(u\ w)P \in \mathcal{P}^u | P = \text{rib-in}(u \Leftarrow w)\}$$

Finally, we define the best possible path that is available to $u$ as

$$\text{SPVP-best(u)} = \text{best(SPVP-choices}(u))$$

```
process spvp(u)
begin
receive P from w ⟶
                begin
                    rib-in(u ⇐ w) := P
                    if rib(u) ≠ SPVP-best(u) then
                    begin
                        rib(u) := SPVP-best(u)
                        for each v ∈ peers(u) do
                        begin
                            send rib(u) to v
                        end
                    end
                end
end
```

Figure 6. SPVP Process at Node $u$ from [Ref. 12] )

This path is the highest ranked path node $u$ can use given the messages that have been received and processed from its peers.

Figure 1 shows how SPVP runs for each node $u \in V$. If there is an unprocessed message from any $w \in$ peers(u), the guard **receive** $P$ **from** $w$ can be activated to receive the oldest unprocessed message that $w$ has sent containing path $P$. If there are multiple links with unprocessed messages, any link may be selected. When the guard is activated the message is deleted from the link and processed in one atomic step according to the code following "$\rightarrow$". The code will store the message in rib-in($u \Leftarrow w$). If the current selected path is no longer the best available path, the code will change the current selected path to be the best available path by executing rib(u) := SPVP-best(u). Finally, it will send this path to all neighbors, $v \in$ peers(u).

We use the exact notation as presented in [Ref. 12] to model how the protocol operates as the system in general. Informally, we describe the network state of the system as all values of rib(u), rib-in($u \Leftarrow w$), and the state of all communication links. The current path assignment at each node implicitly defines a path assignment

for the entire system if $\pi(u) = \text{rib(u)}$.

We model (logical) time t with discrete values $0, 1, 2, ....$ For each node $u$ and each $w \in$ peers(u), $\text{mq}(u \Leftarrow w, t)$ denotes the state of the communication link from node $w$ to node $u$ at time $t$. This is a FIFO message queue, and the notation $\text{mq}(u \Leftarrow w, t)[i]$ refers to the $i$th element of the queue. In particular, $\text{mq}(u \Leftarrow w, t)[1]$ is the first or oldest unprocessed message in the communication link. For each $u$, $\text{rib(u, t)}$ denotes the value of rib(u) at time $t$. For each $u$, and each $w \in$ peers(u), $\text{rib-in}(u \Leftarrow w, t)$ denotes the value of $\text{rib-in}(u \Leftarrow w)$ at time t.

The *network state at time t*, denoted $s(t)$, is comprised of all values $\text{rib(u, t)}$, $\text{rib-in}(u \Leftarrow w, t)$ , and $\text{mq}(u \Leftarrow w, t)$ .

At each state transition from $s(t-1)$ to $s(t)$ either (1) the network state remains unchanged, or (2) some node $u$ processes a message from some $w \in$ peers(u). Note that at each transition, only one node processes a message at a time. We define $\sigma$ as a sequence of nodes, where at each time $t$, a the $t^{th}$ node of the sequence is activated and processes one message. Let $s_0 = s(0)$ be some initial state of path assignments, rib-in's and message queues. We describe $\sigma$ as *fair* with respect to $s_0$ if any message sent from a node $w$ to a node $u$ will eventually be processed.

**Definition: Safe (SPVP)** A stable paths problem is called safe if the protocol SPVP always converges, for any intial state $s_0$ and any fair sequence $\sigma$. If at time t the network state $s(t)$ is such that all message queues $\text{mq}(u \Leftarrow v, t)$ are empty then we say the system has converged at time t, and write $\text{S}(\sigma, s_0, t) \downarrow$.

More detail about SPVP may be found in [Ref. 12].

## 2. Single Node Activation Sequences Model

Several models have been proposed that model BGP behavior that rely on one or more nodes being *activated* at a given point of time. When a node is activated, this abstractly corresponds to the node receiving instantaneous, simultaneous UPDATE messages from all neighbors and selecting the best available path. Feamster *et al.* proposed a BGP model ("Routing protocol dynamics") based on activating only a

single node at a time [Ref. 5]. Feamster also included a framework to describe BGP filtering and ranking. However, the Single Node Activation Sequence Model (SNASM) will be described in terms of the stable paths problem.

**Definition: Infinitely Often** For a sequence of elements $\sigma = a_1, a_2, a_3...$, an element $b$ is said to appear *infinitely often* if the element $b$ is repeated in the sequence infinitely many times.

**Definition: Fair Single Node Activation Sequence** A sequence of nodes $\omega = u_1, u_2, u_3...$ is said to be a *fair single node activation sequence* if each node $u_i \in V$ and appears infinitely often in the sequence.

In order to introduce the SNASM, it will be necessary to redefine some functions in order to introduce the concept of discrete time. We define the path assignment of all nodes at time $t$ as $\pi(t)$ (a mapping from $V$ to $\mathcal{P}$). We define the path assignment at a particular node $u$ at time $t$ as $\pi(u, t)$.

The set of available paths choices($\pi(t)$,$u$,$t$) from node $u$ at a particular time $t$ is defined to be

$$\text{choices}(\pi(t),u,t) = \begin{cases} \{(uv)\pi(v,t) | \{u,v\} \in E\} \cap \mathcal{P}^u & u \neq 0 \\ \{(O)\} & u = 0 \end{cases}$$

Figure 7 presents the SNASM Routing Protocol Dynamics. Time is modeled discretely. The model begins with an initial path assignment at time 0 which is $\pi(0)$. The model uses a fair single node activation sequence to represent the fact that in BGP, each BGP speaker will always be ready to receive and process UPDATE messages from its peers. At each time $t$ a node $u_t$ is activated. This corresponds to the node receiving all the current path assignments of neighbors simultaneously and instantaneously. The node will then pick its highest ranked and available path. Clearly, the model defines a sequence of path assignments $\pi(0), \pi(1), \pi(2), ...$ for each time $t = 0, 1, 2, ...$ This model differs from SPVP because it does not take into account that messages may be in transit, and may be processed in different orders if they are from different neighbors. However, this model can take into account the fact that when a node changes path, the path it changes to may no longer actually be available.

For instance, suppose a node $v$ is being activated and takes the highest ranked path from a node $w$ of the form $(v\ w\ z)P$. It is possible that $z$ has changed its path since $w$ was last activated. While $w$ advertises the path as available, it isn't.

SNASM Routing Protocol Dynamics

At time $t-1$, the current path assignment is $\pi(t-1)$. Each node $u$ has currently selected path $\pi(u, t-1)$ to the destination 0. At time $t$:

1. A given node $u_t$ is activated
2. Node $u_t$ updates its path to be the most preferred and available path which is best(choices($\pi(t-1)$,$u_t$,$t-1$)). Therefore, $\pi(u_t, t) =$ best(choices($\pi(t-1)$,$u_t$,$t-1$)).
3. All other nodes leave their paths unchanged. Therefore, if $v \in V - \{u_t\}$, then $\pi(v, t) = \pi(v, t-1)$

Figure 7. The SNASM Routing Protocol Dynamics

We may now define safety in terms of SNASM.

**Definition: Safe (SNASM)** An instance of the stable paths problem is *safe (SNASM)* if for any initial path assignment $\pi(0)$ and any single node fair activation sequence $u_1, u_2, ...$, there exists a finite $T$ such that $\pi(t) = \pi(T)$ for all $t \geq T$. In Chapter 4 we show that if an instance of SPP is safe (SNASM) this does not imply that it is safe (SPVP).

## 3. Multiple Node Activation Sequence Model

Gao and Rexford also proposed a BGP model in which nodes are activated, and receive the highest ranked path available. However, in their model, multiple nodes may be activated simulatenously. Gao and Rexford also described rankings and filterings in terms of their own framework. However, we will write the multiple node activation sequence model (MNASM) in terms of the stable paths problem.

**Definition: Fair Multiple Node Activation Sequence** A sequence of sets of nodes $\omega = U_1, U_2, U_3...$ is said to be a *fair multiple node activation sequence* if

each node $v \in V$ appears infinitely often in the sequence as the element of some set $U_k \subseteq V$.

MNASM Routing Protocol Dynamics

At time $t-1$, the current path assignment is $\pi(t-1)$. Each node $u$ has currently selected path $\pi(u, t-1)$ to the destination 0. At time $t$:

1. A Set of nodes $U_t$ is activated
2. Each node $v \in U_t$ updates its path to be the most preferred and available path which is best(choices($\pi(t-1)$,$v$,$t-1$)). Therefore, if $v \in U_t$, then $\pi(v, t) =$ best(choices($\pi(t-1)$,$v$,$t-1$)).
3. All other nodes $V - U_t$ leave their paths unchanged. Therefore, if $w \in V - U_t$, then $\pi(w, t) = \pi(w, t-1)$.

Figure 8. The MNASM Routing Protocol Dynamics

Figure 8 presents the Multiple Node Activation Sequence Model. Time is modeled discretely. The model uses a fair multiple node activation sequence, so each node is activated infinitely often. The model begins with a path assignment $\pi(0)$. At each time $t$, a set of nodes $U_t$ are activated. This corresponds to each node in $U_t$ instantaneously and simultaneously receiving the path assignments at time $t-1$ from all other nodes. Each node in $U_t$ will then update its current path assignment to the highest ranked available path. Note that this model differs from SPVP because it does not allow for messages from different neighbors to arrive and be processed in different orders. However, it can model the possibility that the routing information at a given node is not current.

We can now define safety in terms of the multiple node activation sequence model.

**Definition: Safe (MNASM)** An instance of the stable paths problem is safe (MNASM) if for any initial path assignment $\pi(0)$ and multiple node fair activation sequence $U_1, U_2, ...$, there exists a finite $T$ such that $\pi(t) = \pi(T)$ for all $t \geq T$.

### 4. Comparison of Models to BGP

For several reasons, the simple path vector protocol most accurately models BGP behavior. First, RFC 4271 specifies that only one UPDATE message may be processed at any given time. Second, while MNASM can model multiple nodes receiving update messages simultaneously, we will show in Chapter IV that SPVP can match any path assignment reached by MNASM.

However, MNASM is a much simpler model to conduct proofs on because one does not need to keep track of the state of message queues. In Chapter V, we will use MNASM for the proof of the main theorem of this paper.

## E. ROBUSTNESS

**Definition: Robustness** An instance of SPP is *robust (MODEL)* if and only if that instance and every subinstance is uniquely solvable and safe (MODEL).

For this paper, if no model is specified, we take robust to mean robust (MNASM).

## F. DISPUTE WHEELS

The concept of dispute wheels was first introduced by Griffin and Wilfong [Ref. 12]. A dispute wheel is a sequence of nodes and paths that represent mutually conflicting policies due to rankings. These mutually conflicting paths may cause an instance of SPP to be unsolvable or give rise to permanent oscillations, making the instance not safe.

Formally, a *dispute wheel*, $\Pi = (\overrightarrow{U}, \overrightarrow{Q}, \overrightarrow{R})$, of size $k$ is a sequence of nodes $\overrightarrow{U} = u_0, u_1, ...u_{k-1}$, and sequences of nonempty paths $\overrightarrow{Q} = Q_0, Q_1, ...Q_{k-1}$ and $\overrightarrow{R} = R_0, R_1, ...R_{k-1}$, such that for each $0 \leq i \leq k-1$ the following hold true:

1. $R_i$ is a path from $u_i$ to $u_{i+1}$
2. $Q_i \in P^{u_i}$
3. $R_i Q_{i+1} \in P^{u_i}$
4. $\lambda^{u_i}(Q_i) \leq \lambda^{u_i}(R_i Q_{i+1})$

Paths of the form $Q_i$ are often described as spoke paths. Paths of the form $R_i$ are often described as rim paths. Rule 1 specifies the form of a rim path. Rule 2 specifies that that that a spoke path must be available from the originating node, thus assuring that this path can be assigned to the originating node. Rule 3 specifies that the combined rim path and spoke path must be available from the originating node, thus assuring that this path can be assigned to the originating node. Rule 4 stipulates the preference for the combined rim path and spoke path, over a spoke path. Because every node in the sequence $\overrightarrow{U}$ has this property, the policies are mutually conflicting.

Figure 9 presents a generalized dispute wheel. The next section presents specific instances of SPP and examples of their dispute wheels.

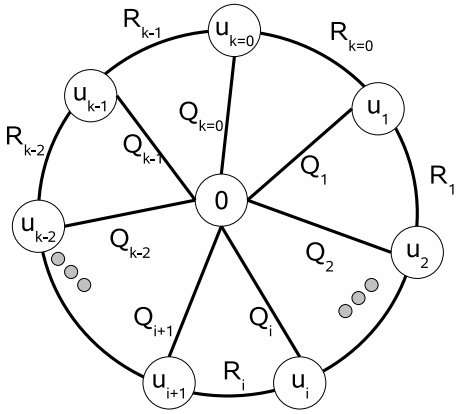


Figure 9. A Generalized Dispute Wheel

As discussed above, a dispute wheel represents a set of mutually conflicting rankings for some nodes. In BGP, this would represent a set of mutually conflicting policies. Griffin and Wilfong proved several theorems about instances of SPP that do not contain dispute wheels [Ref. 12]. To summarize, they proved that if an instance of SPP, $S$ does not contain a dispute wheel, then $S$ is uniquely solvable, safe (SPVP), and robust.

**Theorem V.4 from [Ref. 12] 1.** *If the stable paths problem $S$ has no dispute wheel, then $S$ has a unique solution.*

**Theorem V.9 from [Ref. 12] 1.** *If $S$ has no dispute wheel, then $S$ is safe (SPVP).*

**Theorem V.10 from [Ref. 12] 1.** *Let $S$ be an instance of the stable paths problem. If $S$ has no dispute wheel, then $S$ is robust (SPVP).*

Once Griffin and Wilfong have presented dispute wheels, they describe one set of constraints that can prevent dispute wheels. They show that any instance of SPP that uses route filtering alone, and ranks paths based only on hop count can not contain a dispute wheel. If these constraints are followed for the stable paths problem $S$, then $S$ is guaranteed to be robust.

## G. INTERESTING INSTANCES OF SPP

### 1. Solvable, but not Safe (SNASM or MNASM or SPVP)

In Figure 10, we present an instance of SPP that is solvable, but not safe (SPVP or MNASM). We call this instance "NEXT." NEXT has three solutions. In one solution, $\pi = (1\ 0)(2\ 3\ 1\ 0)(3\ 1\ 0)$. In the second solution, $\pi = (1\ 2\ 0)(2\ 0)(3\ 1\ 2\ 0)$. In the third solution, $\pi = (1\ 2\ 3\ 0)(2\ 3\ 0)(3\ 0)$.

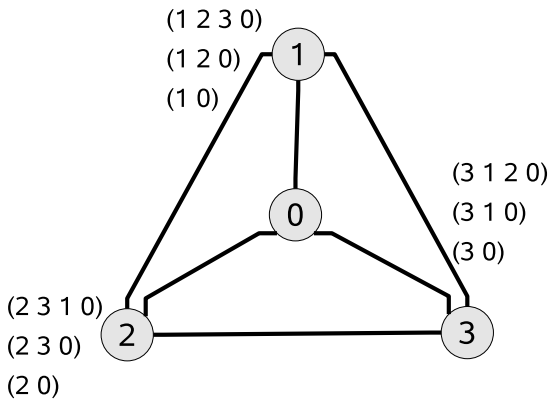


Figure 10. SPP Instance NEXT

Despite having three solutions, there is an initial path assignment and fair multiple node activation sequence that is not safe (SNASM or MNASM). Consider the initial path assignment $\pi(0) = (1\ 0)(2\ 0)(3\ 1\ 0)$. Table I gives an unsafe fair activation sequence for NEXT. This activation sequence could consist of either single

nodes or singleton sets, so both activation models, SNASM and MNASM apply to it. We can see that this sequence of path assignments could repeat indefinitely under a fair activation sequence because the path assignment at time 0 is the same at time 6, and all nodes are activated at least once in between. We claim that NEXT is also not safe (SPVP). In Chapter IV we prove why this is true.

| time $i$ | $\pi(i)$ |
|---|---|
| 0 | (1 0) (2 0) (3 1 0) |
| 1 | (1 2 0) (2 0) (3 1 0) |
| 2 | (1 2 0) (2 3 1 0)(3 1 0) |
| 3 | ((1 2 0) (2 3 1 0) (3 1 2 0) |
| 4 | (1 0) (2 3 1 0) (3 1 2 0) |
| 5 | (1 0) (2 0)(3 1 2 0) |
| 6 | (1 0) (2 0) (3 1 0) |

Table I. Path Assignments of NEXT. If a path assignment is underlined, that node has been activated at that time.

## 2. Uniquely Solvable, but Not Safe (MNASM)

In Figure 11, we present an instance of SPP that is uniquely solvable, but not safe (MNASM). The instance has the unique solution $\pi = (1\ 3\ 0)(2\ 0)(3\ 0)(4\ 3\ 0)$.

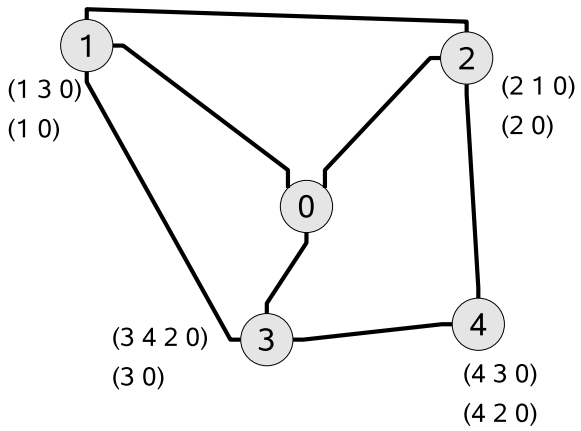


Figure 11. An Instance of SPP that is Uniquely Solvable, but Not Safe (Naughty Gadget from [Ref. 12] )

Despite this unique solution, the instance is not safe. Consider the initial path assignment $\pi = (1\ 0)(2\ 0)(3\ 4\ 2\ 0)(4\ 2\ 0)$. Table II gives an unsafe sequence of path assignments that may be repeated indefinitely.

| step | $\pi$ |
|---|---|
| 0 | (1 0) (2 0) (3 4 2 0) (4 2 0) |
| 1 | (1 0) <u>(2 1 0)</u> (3 4 2 0) (4 2 0) |
| 2 | (1 0) (2 1 0) (3 4 2 0) <u>$\epsilon$</u> |
| 3 | (1 0) (2 1 0) <u>(3 0)</u> $\epsilon$ |
| 4 | (1 0) (2 1 0) (3 0) <u>(4 3 0)</u> |
| 5 | <u>(1 3 0)</u> (2 1 0) (3 0) (4 3 0) |
| 6 | (1 3 0) <u>(2 0)</u> (3 0) (4 3 0) |
| 7 | (1 3 0) (2 0) (3 0) <u>(4 2 0)</u> |
| 8 | (1 3 0) (2 0) <u>(3 4 2 0)</u> (4 2 0) |
| 9 | <u>(1 0)</u> (2 0) (3 4 2 0) (4 2 0) |

Table II. An Unsafe Sequence of Path Assignments for NAUGHTY GADGET from [Ref. 12]. If a path assignment is underlined, that node has been activated at that time.

## 3. Categories

In previous sections, we defined some possible properties of instances of SPP such as robustness, unique solvability, and safety. We would like to categorize these properties in relation to one another. Figure 12 shows how properties of an SPP instance relate to one another, over the space of all SPP instances. In this diagram, we assume all definitions of safety and robustness correspond to the same model. In Chapter IV, we will discuss in more detail how these definitions are related. Note that the absence of a dispute wheel is not a sufficient and necessary condition for robustness. In Chapter V, we will introduce an instance of SPP that is robust, but has a dispute wheel. Also note that safety, implies solvability, because for an instance of SPP to be safe there must exist a solution.

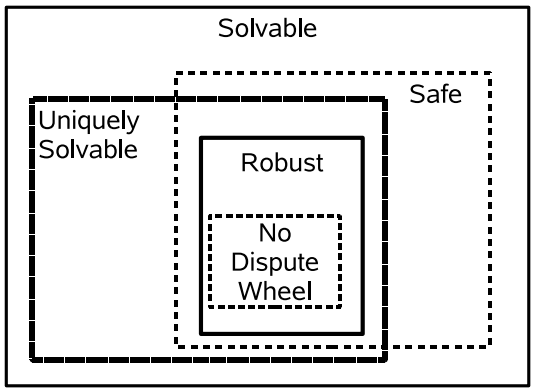


Figure 12. Properties of SPP Instances over the Space of All SPP Instances

## H. HIERARCHICAL BGP

Gao and Rexford [Ref. 8] introduced conditions on filtering, ranking, and topology that guarantee the convergence of BGP. They noted that every eBGP session should define an interorganizational relationship between the two connected ASes. They limited the possible relationships to only peer-to-peer relationships and customer-provider relationships. Therefore, given an AS, $u$, a neighbor $w$ must belong to the set of providers, $provider(u)$; the set of customers $customer(u)$; or the set of peers, $peer(u)$. Note that Gao's definition of $peer(u)$ is much different than Griffin's definition of peers(u). In Gao's definition, a neighbor $w \in peer(u)$ will follow strict guidelines that will be discussed below. In Griffin's definition peers(u) is all the neighbors of $u$, so we have peers(u) = $provider(u) \bigcup peer(u) \bigcup customer(u)$.

Gao and Rexford introduce one topological constraint. There can be no cycle of provider-customer relationships. More precisely let the *provider-to-customer graph* be a subgraph generated where the only edges are directed from provider to customer. This resulting subgraph should be acyclic (or a DAG).

Gao and Rexford introduce a number of filtering policies that reflect the real world configuration of ASes. These policies reflect the idea than an ISP should not advertise routes for traffic without financial benefit. These rules are summarized as the following policies:

- **Exporting to a customer:** In exchanging routing information with a customer, an AS can export its routes, as well as routes learned from its providers and peers.
- **Exporting to a provider or peer:** In exchanging routing information with a provider or peer, an AS can export its routes and the routes of its customers, but it can not export routes learned from other providers or peers.

Gao and Rexford also introduce a number of guidelines on the ranking function of individual ASes. A system of ASes is said to meet *Guideline A [Ref. 8]* if every AS prefers a route via a customer over a route via a provider or peer. Formally, let $S$ be an instance of SPP. For all $u \in V$, for all $P_1, P_2 \in \mathcal{P}^u$ where $P_1 = (ux...0)$ and $P_2 = (uy...0)$, if $x \in customer(u)$ and $y \in provider(u) \bigcup peer(u)$ then $\lambda^u(P_1) > \lambda^u(P_2)$.

Gao and Rexford proved that a system of ASes which follows Guideline A has a stable state and is safe under the Multiple Node Activation Sequence Model. We use a different proof to show that any system of ASes which follows Guideline A can not contain a dispute wheel and is robust.

**Theorem III.1.** *If an instance of SPP meets the exporting policies, the topological constraint and Guideline A from [Ref. 8], then the instance of SPP can't contain a dispute wheel and is robust.*

*Proof.* We use proof by contradiction. Suppose an instance of SPP meets the exporting policies, the topological constraint, and Guideline A [Ref. 8] and has a dispute wheel. Let the dispute wheel $\Pi = (\overrightarrow{U}, \overrightarrow{Q}, \overrightarrow{R})$ have size $k$. For each $Q_j \in \overrightarrow{Q}$ of length $m$, let $Q_j$ be the path $Q_j = (q_{j,0}q_{j,1}q_{j,2}...q_{j,m})$, where $q_{j,0} = u_j$ and $q_{j,m} = 0$. For each $R_j \in \overrightarrow{R}$ of length $n$, let $R_j$ be the path $R_j = (r_{j,0}r_{j,1}r_{j,2}...r_{j,n})$, where $r_{j,0} = u_j$ and $r_{j,n} = u_{j+1}$. Due to the export filters on each AS (or node), for all paths $Q_j$ of size $m$, we must have $q_{j,i-1} \in customer(q_{j,i})$ for all $1 \leq i \leq m$. If this was not the case, then the path $R_{j-1}Q_j$ would not be available to $u_{j-1}$, $R_{j-1}Q_j \notin P^{u_{j-1}}$ due to export filters. Therefore, because $q_{j,0} \in customer(q_{j,1})$, we must also have $r_{j,0} \in customer(r_{j,1})$, otherwise the route $R_{j-1}Q_j$ would not be preferred to $Q_{j-1}$ due to the fact that its first hop would be a provider or a peer. Also due to the export

filters at each node, for all paths $R_j$ of size $n$, we must have $r_{j,i-1} \in customer(r_{j,i})$ for all $1 \leq i \leq n$. If this was not the case then the path $R_jQ_{j+1}$ would not be available to node $u_j$. We have now formed a cycle of customer-provider relationships along the path $(R_0R_1...R_k)$. However, this contradicts the topological constraint that the provider-to-customer graph is acyclic. Therefore, we have a contradiction. We have shown by contradiction that an instance of SPP can never meet the exporting policies, the topological constraint, and Guideline A [Ref. 8] and have a dispute wheel. Therefore, If an instance of SPP meets the exporting policies, the topological constraint and Guideline A [Ref. 8], then the instance of SPP can't contain a dispute wheel. The instance of SPP must also be robust, because it contains no dispute wheels.

□

Gao and Rexford developed this model further to allow for a back-up relationship between neighboring ASes.

# I. CLASS-BASED PATH-VECTOR SYSTEMS

Griffin, Jaggard, and Ramachandran introduced a much more general form of Gao and Rexford's model, called the class-based path-vector system [Ref. 11]. Jaggard and Ramachandran presented a generalized framework that can be used to describe any BGP system where the filtering (also called scoping) rules and ranking rules are based upon the relationships between classes of ASes.

Informally, a *path-vector system* describes some of the low level characteristics of a path-vector protocol. A path-vector system describes the possible destinations, paths to destinations which may be exchanged, rankings for available paths, some basic local import/ export constraints, and some basic import / export transformation rules. Rankings for available paths may be specified similar to the way RFC 4271 ranks paths in BGP, or by use of other metrics such as shortest hop count alone. Basic local constraints make sure that paths known at a given node satisfy certain

properties. For instance, paths should only be imported if the destination is a possible routing destination. Basic / import export rules may be configured to exclude paths which contain loops or perform other, less mundane actions. A path-vector system can be used to describe most characteristics of BGP. However, a path-vector system does not describe exactly how messages are exchanged between nodes.

Griffin, Jaggard, and Ramachandran also define a *policy language* to capture high level characteristics of a system. For BGP, a policy language may describe whether a path is given a specified LOCAL_PREF attribute when the path is imported from specified neighbors. Together, a path-vector system and policy language may be used to describe the stable paths problem.

The class-based path-vector systems are a set of policy languages which meet some general constraints. These constraints are formed using matrices. First, every class-based path-vector system has a set of classes, such as "customer" or "provider". The *cross-class matrix* describes which relationships may occur and row/column numbers correspond to specific classes. Each row and column in this matrix has exactly one "1" and all other entries are "0." This matrix may describe facts such as "customer-provider relationships are allowed" or "customer-peer relationships are not allowed." The *preference matrix* describes some ranking rules for different classes, such as "prefer all paths received from customers to all paths received from providers." The *level matrix* describes the scoping rules such as "export all routes learned from a customer to a provider." These preference matrix and level matrix can also be used to describe hierarchical properties of BGP. For instance, depending on whether a relationship is with a tier 1 or tier 2 peer different exporting rules may be specified.

Jaggard and Ramachandran continued work on class-based path-vector systems by giving a much more general form of Theorem III.1. In their paper they give an exact condition for dispute wheel creation based upon the particular relationships, scoping rules, and ranking rules of a particular system, as well as global constraints [Ref. 18]. This exact condition is still stricter than a necessary and sufficient condi-

tion to guarantee BGP robustness, because as we have seen, some instances of SPP (and systems of ASes) may have dispute wheels and still be robust.

THIS PAGE INTENTIONALLY LEFT BLANK

# IV. COMPARISON OF BGP MODELS

In the previous chapter we introduced three different models of BGP behavior; the simple path vector protocol (SPVP), the single node activation sequence model (SNASM), and the multiple node activation sequence model (MNASM). Because all models are expressed in terms of the stable paths problem, solvability and unique solvability is equivalent between the three different models. However, in this chapter we investigate two other issues. First, we investigate whether any sequence of path assignments given by one model can match another model. Second, we investigate whether safety in one model implies safety in another model.

## A. MATCHING PATH ASSIGNMENTS

We investigate whether any sequence of path assignments given by one model can be matched by another model. Informally, we describe matching as the ability of one model to begin with the same path assignment as another model, and reach all possible subsequent path assignments for the other model. We allow intermediate path assignments to be taken between equal path assignments. We say that the sequence of path assignments $\omega = \pi_1(0), \pi_1(1), \pi_1(2), ...$ matches the sequence of path assignments $\sigma = \pi_2(0), \pi_2(1), \pi_2(2), ...$ if there exists a subsequence of $\omega$ that is equal to $\sigma$.

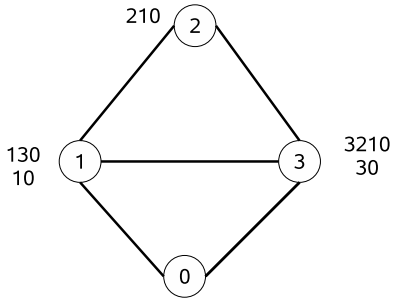


Figure 13. An Instance of SPP That Shows MNASM Does Not Match SPVP

**Definition: Matching of Models** We say that BGP model A *matches* BGP model B, if for any sequence of path assignments given by model B denoted by $\sigma$, there exists a sequence of path assignments given by BGP model A that matches $\sigma$.

We proceed to our negative results first.

**Theorem IV.1** (MNASM does not match SPVP)**.** *The multiple node activation sequence model does not match the simple path vector protocol*

*Proof.* Consider the following counterexample presented in Figure 13. Let $\omega = \pi_\omega(0), \pi_\omega(1), \pi_\omega(2), ...$ be the sequence of path assignments given by SPVP that we will show can not be matched by the multiple node activation sequence model. As usual, the path assignment at node $u$ at time $t$ is denoted by $\pi_\omega(u, t)$. We induce the initial state as follows, let $\pi_\omega(3, 0) = (3\ 0)$ and let nodes 1 and 2 have the empty path assignment. For each $u \in V$ and $w \in$ peers(u), let mq($u \Leftarrow w$) be a message informing $u$ of $\pi(w, 0)$. As depicted in Figure 14, there is a sequence of path assignments in SPVP that gives a final, stable path assignment $\pi_\omega = (1\ 0)(2\ 1\ 0)(3\ 2\ 1\ 0)$. This is achieved by processing messages in the following order. At $t = 1$, node 1 processes mq$(1 \Leftarrow 0)[1]$ and changes its path such that $\pi(1, 1) = (1\ 0)$. At $t = 2$, node 2 processes mq$(2 \Leftarrow 1)[1]$ and keeps the empty path assignment. At $t = 3$, node 2 processes mq$(2 \Leftarrow 1)[1] = (1\ 0)$ and changes its path assignment to $\pi(2, 3) = (2\ 1\ 0)$. At $t = 4, 5, 6, 7$, node 3 processes mq$(3 \Leftarrow 1)[1]$, mq$(3 \Leftarrow 1)[2]$, mq$(3 \Leftarrow 0)[1]$,and mq$(3 \Leftarrow 2)[1]$ and does not change its path assignment. At $t = 8$, node 3 processes mq$(3 \Leftarrow 2)[1] = (2\ 1\ 0)$ and changes its path assignment to $\pi(3, 8) = (3\ 2\ 1\ 0)$.

However, for this same initial path assignment, the multiple node activation sequence model can not reach these subsequent path assignments. This is because when any node is activated, it receives the highest ranked paths. Therefore, only node 1 can change path assignments and will change its path to (1 3 0). This is a stable path assignment, and no future changes can occur.

Therefore, there exists a sequence of path assignments given by SPVP that can't be matched by any sequence of path assignments given by MNASM

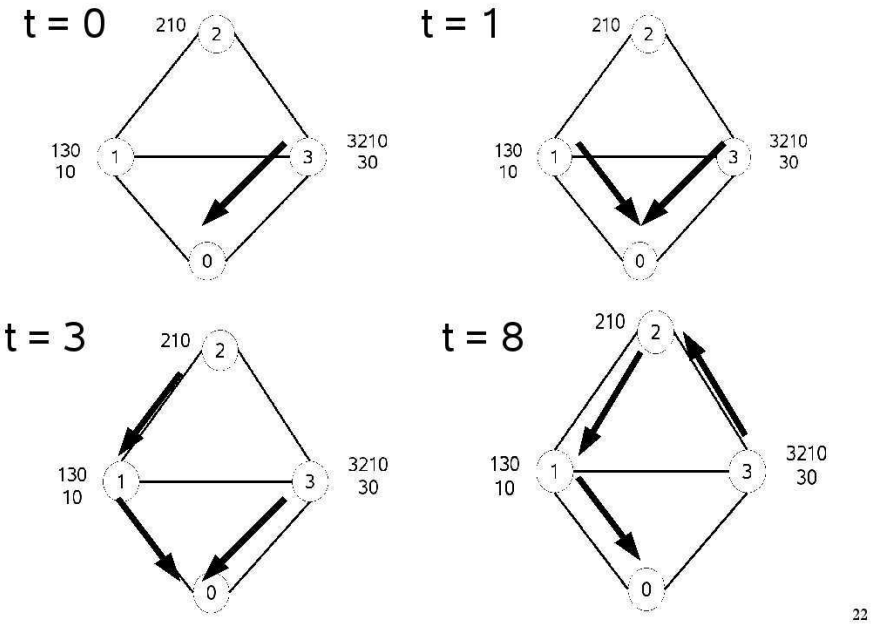


Figure 14. A Sequence of Path Assignments Given by SPVP for Figure 13

We have shown that the multiple node activation sequence model does not match the simple path vector protocol.

□

However, we would like to know whether SPVP matches the multiple node path vector protocol.

**Theorem IV.2** (SPVP matches MNASM)**.** *The simple path vector protocol matches the multiple node activation model.*

*Proof.* Let $S$ be an instance of the stable paths problem. Let $\omega = U_1, U_2, ...$ be any fair multiple node activation sequence and let $\sigma = \pi_\omega(0), \pi_\omega(1), \pi_\omega(2)$ be the sequence of path assignments for $\omega$ given by the MNASM. We would like to show that there exists a sequence of path assignments given by SPVP that matches $\omega$. Let $\theta = \pi_{SPVP}(0), \pi_{SPVP}(1), \pi_{SPVP}(2), ...$ be the subsequence of path assignments given by SPVP we are trying to form. We would like to show that there exists an initial state for SPVP and ordering of message receipts such that $\pi_{SPVP}(i) = \pi_\omega(i)$ for all

$i \geq 0$. Let the initial state be induced by $\pi_\omega(0)$ such that $\pi_\omega(0) = \pi_{SPVP}(0)$, each rib-in$(u \Leftarrow w) = \epsilon$, and each message queue mq$(u \Leftarrow w) = \pi_\omega(w, 0)$.

Let $X(i)$ be the induction predicate that after the $i^{th}$ element of the subsequence $\theta$ has been formed, $\pi_\omega(i) = \pi_{SPVP}(i)$ and for all nodes $u \in V$, if best(choices($\pi(i)$,$u$,$i$)) has a next hop of $w$ and $\pi_{SPVP}(u, i) \neq$ best(choices($\pi(i)$,$u$,$i$)) either there is message in the queue mq$(u \Leftarrow w)$ that informs $u$ of the path best(choices($\pi(i)$,$u$,$i$)) OR rib-in$(u \Leftarrow w)$ has this path stored already.

Base Case. The predicate $X(0)$ holds true because our initial state has those properties.

Induction Step. Suppose $X(i)$ is true. Under MNASM, a set of nodes $U_{i+1}$ will be activated at time $i+1$. For each $u \in U_{i+1}$, we will never process messages that have been generated since the path assignment $\pi_{SPVP}(i)$ was reached, because this may cause a node to take a path assignment other than the one we would like to be taken at $\pi_\omega(i)$. However, we process all other messages in all queues, in any arbitrary order. This will give $u$ an exact picture of what neighboring path assignments were under $\pi_{SPVP}(i)$, and guarantees that the best(choices($\pi(i+1)$,$v$,$i+1$)) will be the final path selected. Once this has been completed for each node $u \in U_{i+1}$, we have created a path assignment $\pi_{SPVP}(i+1) = \pi_\omega(i+1)$ Now, suppose any other node $v$ is no longer assigned the path best(choices($\pi(i+1)$,$v$,$i+1$)) and this path's first next hop is $w$. Suppose the node $v \notin U_i$ . It must still either have a message in the queue mq$(v \Leftarrow w)$ informing $v$ of that path or rib-in$(v \Leftarrow w)$ has this path. Suppose the node $v$ was activated in the set $U_i$. There are two possible cases, or a combination of both. In the first case, a neighboring node $x \in U_i$ was activated and changed its path assignment so now, $v$ has an even higher ranked path available in choices($\pi(i+1)$,$v$,$i+1$) that was not available in choices($\pi(i)$,$v$,$i$). However this path should have been advertised, so there must be a message in the queue mq$(u \Leftarrow x)$ informing $v$ of this path. Otherwise, suppose the path $\pi_{SPVP}(v, i+1)$ has been withdrawn. The new path best(choices($\pi(i+1)$,$v$,$i+1$)) must either be contained in

a $rib - in$ or have been advertised by the withdrawal. Therefore $X(i) \Rightarrow X(i+1)$

By the principle of induction, we've shown that the path assignments generated under MNASM can be generated by taking a subsequence of path assignments under SPVP.

Finally, we must check that the processing of messages is *fair*; every message will eventually be processed. The processing of messages is fair, because every node is activated infinitely often under MNASM, and each time a node is activated, all old messages will processeded before the last patest assignment was generated.

□

**Corollary IV.3** (SPVP matches SNASM)**.** *By a similar argument, the simple path vector protocol matches the single node activation sequence model.*

**Theorem IV.4** (MNASM matches SNASM)**.** *The multiple node activation sequence model matches the single node activation sequence model*

*Proof.* Let $\omega = u_0, u_1, u_2, ...$ be any fair single node activation sequence. We form a fair multiple node activation sequence $\omega^l$ by simply taking one element subsets such that $\omega^l = \{u_0\}, \{u_1\}, \{u_2\}, ...$ . The path assignment for $\omega$ will exactly equal the path assignment for $\omega^l$, because nodes are activated identically under both models, and the same node is activate at each time.

□

Figure 15 depicts the result of this section. We consider the space of instances of SPP and initial path assignments. The intersection of two models describes an instance of SPP and initial path path assignments for both models that match each other for any possible sequence of path assignments. Likewise, the places where model A does not intersect with model B describes an instance of SPP and initial path assignment that results in a sequence of path assignments that can not be matched by model A.

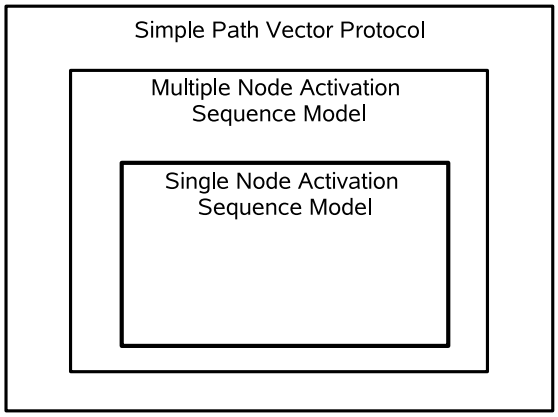


Figure 15. How Models Match Each Other over the Space of Instances of SPP and Initial Path Assignments

## B. COMPARISON OF SAFETY

In the preceeding section, we investigated whether different models of BGP match each other. We can use these results to show that safety as defined in one model can imply safety in another model.

**Theorem IV.5.** *Let $S$ be an instance of the stable paths problem. If $S$ is not safe (MNASM), then $S$ is not safe (SPVP).*

*Proof.* Suppose $S$ is not safe (MNASM). Then there exists at least one fair multiple node activation sequence, such that there is not finite time $T$ where the $\pi(T) = \pi(t)$ for all $t \geq T$. By Theorem IV.2, there is an ordering of message processing such that SPVP will have a subsequence of path assignments with the exact same path assignments. Therefore, the message queues can never empty and $S$ is not safe (SPVP)

□

**Corollary IV.6** (Safe (SPVP) $\Rightarrow$ Safe (MNASM))**.** *By the contrapositive of Theorem IV.5, If $S$ is an instance of SPP that is safe (SPVP), then $S$ is safe (MNASM)*

**Theorem IV.7.** *Let $S$ be an instance of the stable paths problem. If $S$ is not safe (SNASM), then $S$ is not safe (MNASM)*

*Proof.* Suppose $S$ is not safe (SNASM). Then there exists some fair single node activation sequence such that there is no finite time $T$ where where the $\pi(T) = \pi(t)$ for all $t \geq T$. By Theorem IV.4, we can form a fair multiple node activation sequence from this sequence which also has the property where there is no finite time $T$ where where the $\pi(T) = \pi(t)$ for all $t \geq T$. By definition, $S$ is not safe (MNASM).

□

**Corollary IV.8** (Safe (MNASM) $\Rightarrow$ Safe (SNASM))**.** *By the contrapositive of Theorem IV.7, If $S$ is an instance of SPP that is safe (MNASM), then $S$ is safe (SNASM)*

The following corollary is derived from applying Corollaries IV.6 and IV.8.

**Corollary IV.9.** *If $S$ is an instance of SPP that is safe (SPVP), then it is safe (SNASM).*

**Theorem IV.10** (Safe (SNASM) $\not\Rightarrow$ Safe (MNASM))**.** *Let $S$ be an instance of SPP. If $S$ is Safe (MNASM), then this does not imply that $S$ is safe (SNASM).*

*Proof.* Consider the the following counterexample, which is presented in Figure 16. This instance of SPP is not safe (MNASM). Let $S$ have the initial path assignment $\pi(1, 0) = (10)$ and $\pi(2, 0) = (20)$. This routing system will not converge under the fair multiple node activation sequence, $\{12\}, \{12\}, \{12\}, ....$ However, $S$ is safe (SNASM). Given any fair single node activation sequence and initial path assignment, it will always converge.

□

Figure 17 depicts the result of this section. We consider the space to be the set of all instances of the stable paths problem. We are not sure whether Safe (MNASM) $\Rightarrow$ Safe (SPVP) or not. It is possible that these two areas are exactly equal.

In this section we have shown that models of BGP do not necessary have equivalent definitions of safety, and that some path assignments of some models can not necessarily be matched by other models. These results have important consequences. For instance, a theorem proved about robustness using one model, may not necessarily imply robustness for other models.

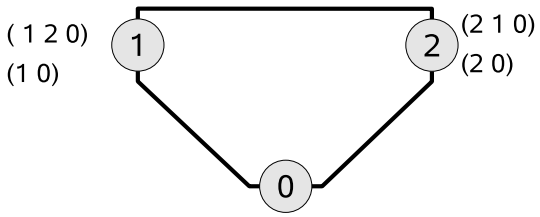


Figure 16. Instance of SPP for Theorem IV.10. DISAGREE from [Ref. 12]

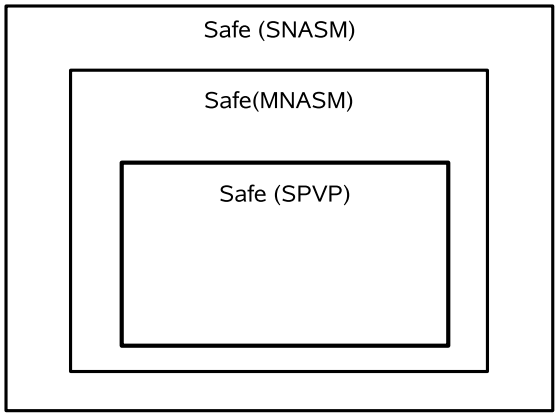


Figure 17. Safety Between Different Models

# V. A WEAKER SUFFICIENT CONDITION FOR ROBUSTNESS

## A. MOTIVATION

There have been several proposals to guarantee the robustness of BGP. In this paper, we pursue an approach that relies on operational guidelines and global constraints. Griffin *et al* showed that if an instance of SPP does not have a dispute wheel, the instance must be robust. Unfortunately, this condition is too strict; there exist instances of SPP which contain dispute wheels but are robust. Consider the instance of SPP in Figure 18. This instance of SPP is robust, but contains the dispute wheel in Figure 19. In this section, we give a weaker sufficient condition for robustness. Our approach focuses on determining whether the subinstance of SPP generated for each dispute wheel 1) is robust and 2) has the property such that for each node of the dispute wheel, all possible paths are contained in the dispute wheel.

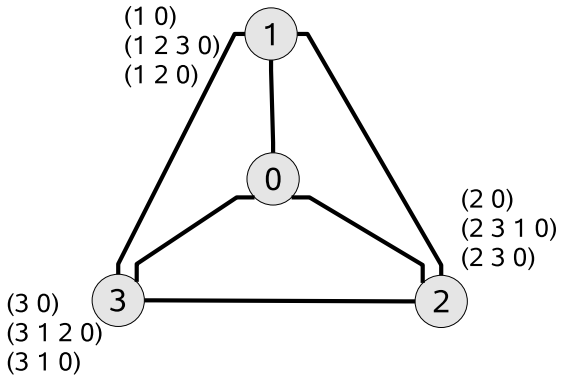


Figure 18. An Instance of SPP that has a Dispute Wheel, but is Robust

Once we give a weaker sufficient condition for robustness, we investigate how to determine whether instances of SPP are robust, despite the presence of one or more dispute wheels. We focus on developing new global and local constraints that guarantee robustness, despite the presence of a dispute wheel.

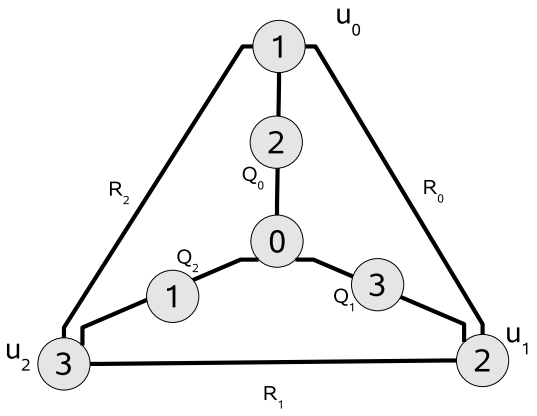


Figure 19. The Dispute Wheel of Figure 18

## B. SUBINSTANCES OF SPP FROM DISPUTE WHEELS

Given an instance of SPP $S = (G, \mathcal{P}, \Lambda)$, there is a natural way to derive subinstances of SPP from the dispute wheels of $S$. Given a dispute wheel $\Pi = (\overrightarrow{U}, \overrightarrow{Q}, \overrightarrow{R})$, we define SPP($\Pi$) = $(G_\Pi, \mathcal{P}_\Pi, \Lambda_\Pi)$ to be the derived instance of SPP from $\Pi$. Let the graph $G_\Pi = (V_\Pi, E_\Pi)$ have the property where $V_\Pi$ contains every vertex that appears in $\overrightarrow{Q}$ and $\overrightarrow{R}$ and $E_\Pi$ contains every edge $(u\ v) \in E$ such that $u, v \in V_\Pi$ . Let the set of available paths $P_\Pi = \{P | P \in \mathcal{P}\}$ and every edge in the path $P$ is present in $E_\Pi$ . For each node $u$, we will denote its set of available paths as $\mathcal{P}^u_\Pi$. Finally, let the ranking function be $\Lambda_\Pi = \Lambda$, but modified to exclude all omitted paths.

We define a dispute wheel $\Pi$ to be *robust* if SPP($\Pi$) is robust.

## C. ALL DISPUTE WHEELS ROBUST IMPLIES UNIQUELY SOLVABLE

To prove than an instance of SPP is robust, we need to show that the instance (and every subinstance) of SPP is uniquely solvable.

**Theorem V.1.** *If every dispute wheel of a stable paths problem is robust (or even just uniquely solvable), then the stable paths problem is uniquely solvable.*

This proof closely follows the proof of Theorem V.4 [Ref. 12].

*Proof.* We use proof by contradiction. Let $S$ be an instance of the stable paths problem. Suppose that every dispute wheel of $S$ is robust, and it has at least two distinct solutions $\pi_1 = (P_1, ..., P_{n-1})$ and $\pi_2 = (Q_1, ...Q_{n-1})$. As discussed above, every solution defines a tree rooted at the origin. Let $T_1$ and $T_2$ be trees, rooted at the origin the origin, that are defined by $\pi_1$ and $\pi_2$ respectively. Given a graph or component $G$ let $V(G)$ and $E(G)$ be the vertices and edges of the graph or component respectively. Let $H$ be the graph $(V, E(T_1) \cap E(T_2))$. Let $T$ be the connected component of $H$ containing the origin. Note that $T$ must be a tree because it is a intersection of two trees. Every edge of $E(T_1 \cup T_2)$ not contained in $E(T) = (T_1 \cap T_2)$ is either in $E(T_1 - T_2)$ or $E(T_2 - T_1)$. We say an edge $(uv)$ is entering a set of vertices $V$ if exactly one of the nodes $\{u, v\}$ is in the set of vertices $V$. Therefore, every edge of $T_1 \cup T_2$ entering $V(T)$ must either be in $E(T_1 - T_2)$ or $E(T_2 - T_1)$.

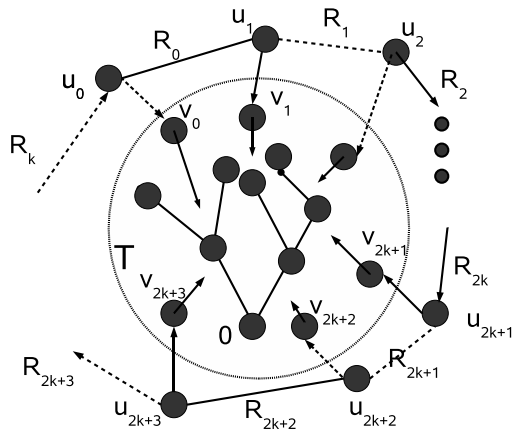


Figure 20. Illustration for Theorem V.1. The nodes inside the dashed circle all represent nodes belonging to $T$. Outside the circle are edges and nodes in $T_1$ and $T_2$. Dashed Edges are in $T_2$. Solid edges are in $T_1$

We now construct a dispute wheel. Figure 20 visually presents the dispute wheel that will be formed. Note that because the two solutions are unique, $T_1 \neq T_2$, the set of vertices $V - V(T)$ must be nonempty and at least one of the trees has an edge entering $V(T)$. Without loss of generality, consider any two nodes $u$ , $v$ in $T_1$ such that $v \in T$ and $u \notin T$. The node $u$ can not have the empty path assignment in $\pi_2$ because

it can not prefer the empty path to the available path $(uv)P_v$. Therefore, the node $u$ must also belong to $T_2$. We will begin to construct the dispute wheel by choosing an edge $\{u_0, v_0\} \in T_1$ such that $u_0 \notin V(T)$ and $v_0 \in V(T)$. As discussed above $u_0$ does have a path to the origin through $T_2$ which must be of the form $R_0(u_1v_1)Q_{v_1}$ that has the following properties: (i) $u_1 \notin V(T)$ and $v_1 \in V(T)$ (ii) The path $R_0$ is a path from $u_0$ to $u_1$ in $T_2$ and contained entirely in the node set $V - V(T)$ (iii) Finally, $R_0$ must have a length of at least one, otherwise one of the paths $\pi_1(u_0)$ or $\pi_2(u_0)$ would be unstable. This process is repeated at node $u_1$ except now we already have a path directly to the origin for $T_2$ and we are looking for a path to the origin through $T_1$. We continue alternating and searching for paths in this fashion until we eventually repeat some node, which without loss of generality is $u_0$. We must eventually repeat a node because the set of nodes in $V - V(T)$ is finite and during our search we continue to reach a new node each time unless a node has been repeated.

We must now show that we have created a dispute wheel. Due to our construction, we have already shown all the properties of a dispute wheel except that for each $i$, $\lambda^{u_i}((u_iv_i)Q_i) \leq \lambda^{u_i}(R_i(u_{i+1}v_{i+1})Q_{i+1})$. To show this, we assume without loss of generality that the path $(u_iv_i)Q_i$ is contained in $T_1$ . Suppose the inequality did not hold. Then we would have $\lambda^{u_i}((u_iv_i)Q_i) > \lambda^{u_i}(R_i(u_{i+1}v_{i+1})Q_{i+1})$ which would mean that $T_2$ should have preferred the same path to $T$ and that $T_2$ is not stable. But, this contradicts our assumption, so the inequality must hold and we must have created a dispute wheel that has at least two distinct solutions.

However, the dispute wheel must have a unique solution, because every dispute wheel of $S$ is robust. Therefore, we have a contradiction. We have used indirect proof to show that if every dispute wheel of an instance $S$ of SPP is robust, then the instance of SPP is uniquely solvable.

□

# D. ALL DISPUTE WHEELS ROBUST AND COMPLETE IMPLIES SAFETY

To prove an instance of SPP is robust, we must also show that the instance (and every subinstance) is safe.

Griffin *et al.* gave a procedure to construct a dispute wheel given an unsafe instance of SPP. We will use a similar method to construct a dispute wheel that is not safe. They used the procedure to prove the following theorem. [Ref. 12]:

**Theorem V.9 from [Ref. 12] 2.** *If $S$ has no dispute wheel, then $S$ is safe (SPVP).*

## 1. Selecting an Appropriate Model

In the previous chapter, we compared the various BGP Models. For the following proofs, we will use the multiple node activation sequence model. We believe that these results could be proved differently to provide similar results for the simple path vector protocol. However, for the remainder of the chapter, when we describe a stable paths problem to be "safe," we specifically mean that it is safe (MNASM). Likewise, when we describe a stable paths problem to be "unsafe," we mean that it is not safe (MNASM).

## 2. Complete Dispute Wheels

We introduce the concept of a *complete* dispute wheel. We will show that if every dispute wheel of an instance of SPP is complete and robust, then the instance is robust. Much of the following notation is taken from [Ref. 12].

Suppose $S$ is an instance of the stable paths problem that is not safe (MNASM). For some initial path assignment $\pi(0)$ and activation sequence $\sigma$, there does not exist any finite time $T$ such that the path assignment does not change after time $T$. However, there exist some nodes that do not change their paths infinitely often. We define the set of nodes $\mathcal{C}$ to be the nodes that do not change their path assignment after time $T_c$. We define the set $\mathcal{O}$ to be the set of nodes that change their paths infinitely often. For each node $u \in V$ we define values($\sigma$,$\pi(0)$,$u$) to be the set of paths

that $u$ adopts infinitely often. Note that for $u \in \mathcal{C}$, values($\sigma$,$\pi(0)$,$u$) will be a one element set, equal to $\{\pi(t_c, u)\}$.

Suppose $P$ is a path of the form $(w_0 w_1 ... w_k)$. We define $P[w_i w_j]$ as the subpath $(w_i w_{i+1} ... w_j)$. Also, we define $P[1]$ to be the first node $w_0$.

Let $S$ be an unsafe instance of SPP. Let $U$ be the set of all nodes such that $u \in \mathcal{O}$ and $u$ adopts a path $(uw)Q \in$ values($\sigma$,$\pi(0)$,$u$) such that $w \in \mathcal{C}$. For any node $u \in U$, let Q-path($u$) be the lowest ranked path of values($\sigma$,$\pi(0)$,$u$) that goes directly to $\mathcal{C}$. Finally, we define RQ-paths($u$) to be the set of paths values($\sigma$,$\pi(0)$,$u$) $-$ {Q-path($u$)}. By Lemma V.6, if $P \in$ RQ-paths($u$), we can write this path as $P = R$ Q-path($v$) where $R$ is a path of the form $(u\ w_1\ w_2 ... v)$ where $v \in U, w_i \notin U$ and Q-path($v$) is a path that leads directly to some fixed node $w \in \mathcal{C}$. We denote the set of all paths of the form $(u\ w_1\ w_2 ... v)$ as R-paths($u$). Note that for each path of RQ-paths($u$), there is a corresponding subpath in R-paths($u$). Finally, for such a path $P$, we define entering($P$) as the node $v$ that enters $\mathcal{C}$ by routing through $w$. Some of this terminology is presented in Figure 21.

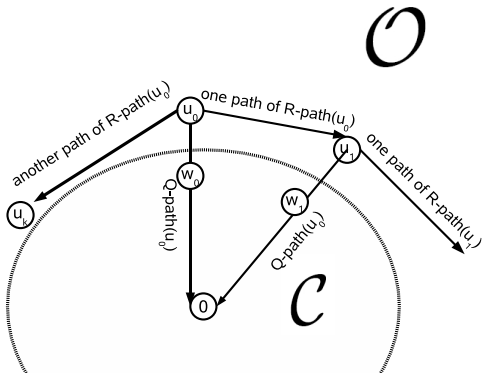


Figure 21. An Illustration of Some Terminology

**Definition: Complete Dispute Wheel** Let $S$ be an instance of SPP. We define a dispute wheel of to be *complete* and denote it $\Theta(S)$ if for all $u \in \Theta(S)$, we have $\mathcal{P}^u{}_{\mathrm{SPP}(\Theta(S))} = \mathcal{P}^u_S$. (for each node in the dispute wheel, all available paths for the instance $S$ are contained inside the dispute wheel)

## 3. Existence of Dispute Wheel for an Unsafe Instance of SPP

The following lemmas will be needed to show that every unsafe instance of SPP contains a dispute wheel.

**Lemma V.2.** *If every node of a digraph $G$ has an outgoing degree of exactly one, the graph must contain a cycle.*

*Proof.* We will show that this is true by induction on the number of nodes of the graph, which will be denoted by $i$. There are also exactly $i$ edges. Let $W(i)$ be the induction hypothesis that every graph with $i$ nodes contains a cycle if the graph has $i$ edges.

Base Case. Let $i = 1$. This one element graph contains a cycle, because any outgoing edge from the single node must be to itself. Therefore $W(1)$ is true.

Induction Step. Suppose every graph with $i$ nodes and edges contains a cycle. Let $G$ be some arbitrary graph with $i + 1$ nodes and $i + 1$ edges. Let $u$ be some arbitrary edge. If $u$ has an outgoing edge to itself, then $G$ must contain a cycle and $W(i + 1)$ is true. Otherwise suppose, $u$ has an outgoing edge to some node $v$.

In our first case, suppose $u$ has no incoming edges. If we remove $u$ and the outgoing edge $(u\ v)$, we will be left with a graph with exactly $i$ nodes and edges. This graph must have a cycle, so $G$ must have a cycle and $W(i + 1)$ must be true.

In our second case, suppose $u$ has one or more incoming edges from nodes $x, y, z....$ For each such edge $(x\ u), (y\ u), (z\ u)...$, we replace the edge with $(x\ v), (y\ v), (z\ v)....$ Finally, we remove node $u$ and the edge $(u\ v)$. We are left with a graph with $i$ nodes and $i$ edges. This graph must contain a cycle. This implies that $G$ must have also contained a cycle. We have examined all cases and $W(i) \Rightarrow W(i + 1)$.

By the principle of induction, if every node of a digraph $G$ has an outgoing degree of exactly one, the graph must contain a cycle.

□

Now, suppose instead of having an outgoing degree of exactly one, a graph has an outgoing degree of one or more. The graph must still contain a cycle, because we are introducing additional edges.

A strongly connected component is a maximal subgraph of a digraph such that each element of the subgraph can reach every other element of the subgraph.

**Lemma V.3.** *Let $G = (V, E)$ be a digraph. $G$ has at least one strongly connected component with no outgoing edges to other strongly connected components.*

*Proof.* We use proof by contradiction. Suppose every strongly connected component had at least one outgoing edge to another strongly connected component and there are $n$ strongly connected components. Every digraph may be decomposed completely into strongly connected components, creating another digraph of strongly connected components [Ref. 3] . If every connected component has at least one outgoing edge to another strongly connected component, there must be a cycle of strongly connected components by Lemma V.2. However, this reaches a contradiction because the cycle of connected components would itself be a larger connected component. Therefore, There must be at least one strongly connected component with no outgoing edges to other strongly connected components.

□

For any unsafe instance of SPP, we show how a dispute wheel $\Pi$ may be created. It is possible that more than one dispute wheel may be generated by the procedure. A similar proof was given from [Ref. 12] as Theorem V.9.

A *closed walk* is a path on a component or graph such that the path visits every node and edge at least once, and begins and ends with the same node.

**Lemma V.4.** *Let $S$ be an unsafe instance of SPP. $S$ has a dispute wheel.*

*Proof.* Let $\mathcal{P}$ be the set of paths which contains the every path $P \in \{\text{R-paths}(u) | u \in U\}$. Let $G(\mathcal{P})$ be the graph induced by taking all the edges and nodes of the paths of $\mathcal{P}$. Each node $u$ has one or more paths in R-paths($u$) and must have an outgoing degree of one or greater in $G(\mathcal{P})$. Furthermore, consider any node

v along a path R-paths($u$) such that $v \in \mathcal{O} - \mathcal{U}$. This node must also have an outgoing degree of at least one in $G(\mathcal{P})$, because it is along a path, and it can't be the last node of the path, because that node is in $U$. By Lemma V.3, $G(\mathcal{P})$ must contain at least one strongly connected component that contains no outgoing edges to other strongly connected components. Furthermore, this strongly connected component must contain at least one node $u \in U$. This is because if it contains a node $v \in \mathcal{O} - \mathcal{U}$, this node will have a path to a node in $u$, which must belong to the same strongly connected component. Let $C$ be such a strongly connected component

We claim that we can generate a dispute wheel from $C$. If we conduct a closed walk on the resulting graph, we will have visited each node $u \in C \cap U$ and each path $P \in \{\text{R-paths}(u) | u \in C \cap U\}$ at least once. We form our dispute wheel by beginning the walk with an arbitrary node $u \in C \cap U$ which we take to be $u_0$. We take $Q_0 = \text{Q-path}(u_0)$ and $R_0$ to be the path of R-paths($u$) we take first. For each subsequent node $v \in C \cap U$, if this is the i$^{th}$ time we have reached a node in $C \cap U$, we take $u_i = v$, $Q_0 = \text{Q-path}(u_k)$, and finally, we take the next path traveled to be $R_i$. This process terminates at the end of our closed walk. We take the last node reached $u$ to be $u_k = u_0$.

We must now show that we have generated a dispute wheel. Clearly, properties 1-3 of a dispute wheel have been satisfied, otherwise those paths would not occur infinitely often. Finally, for any paths $R_i Q_{i+1}$ and $Q_i$, $\lambda^{u_i}(Q_i) \leq \lambda^{u_i}(R_i Q_{i+1})$, because otherwise node $u_i$ would never switch paths away from $Q_i$ because that path is always available.

□

## 4. All Dispute Wheels Robust and Complete Implies Safety

In the following lemma, we show that there is a time where all paths that do not occur infinitely often, can no longer be path assignments for any node.

**Lemma V.5** (Flushing Paths That Do Not Occur Infinitely)**.** *Let $S$ be an unsafe instance of the stable paths problem. Let $w$ be a node in $V$. Suppose that $P \notin$ values($\sigma$,$\pi(0)$,$w$). Then there is a time $t_f$ after which no finite path of the form $QP$ belongs in $\pi(t)$.*

*Proof.* We use proof by induction on the length of $Q$. Let $Z(i)$ be the predicate that there exists a time $t_i$ after which no path of the form $QP$ belongs in $\pi(t_i)$ such that $Q$ has length $i$ .

Base Case. Let $i = 0$. After time $t_c$, the node $w$ can only update its assignment to a path in values($\sigma$,$\pi(0)$,$w$). Node $w$ is activated infinitely often. Let $t_0 = t_w > t_c$ be the next time node $w$ is activated which is after the time $t_c$. After time $t_w = t_0$ there can be no path of the form $QP$ such that $Q$ has length 0. $Z(0)$ is true.

Induction Step. Suppose $Z(i)$ is true. There exists a time $t_i$ after which no path of the form $QP$ belongs in $\pi(t_i)$ such that $Q$ has length $i$. Let $v$ be a node such that $\pi(t_i, v) = QP$ where $Q$ has length $i+1$. Node $v$ is activated infinitely often. Let $t_v > t_i$ be the next time node $v$ is activated. After time $t_v$ there can be no path of the form $QP$ because $v$ can no longer adopt this path. For all $v$ such that $\pi(t_i, v) = QP$ where $Q$ has length $i+1$, let $t_{i+1}$ be $\max(\mathrm{t_v})$. After time $t_{i+1}$ there can be no path of the form $QP$ such that $Q$ has length $i+1$. $Z(i) \Rightarrow Z(i+1)$.

Our predicate $Z(i)$ is true for all $i \geq 0$. By the principle of induction, we have shown that there exists a time $t$ after which no finite path of the form $QP$ belongs in $\pi(t)$.

□

In this theorem, we show that for any path that occurs infinitely often, all subpaths must also occur infinitely often.

**Lemma V.6.** *For some node $u$, if $P \in$ values($\sigma$,$\pi(0)$,$u$) where $P = (w_0 w_1 ... w_k)$ , then for all $w_i$, $P[w_i 0] \in$ values($\sigma$,$\pi(0)$,$w_i$).*

*Proof.* If $P[w_i 0] \notin$ values($\sigma$,$\pi(0)$,$w_i$) and $P \in$ values($\sigma$,$\pi(0)$,$u$) there would be a contradiction, because by V.5 the path $P$ should have been flushed after some time $t$.

□

In the following theorem we give our main result of this subsection. We show that for an instance of SPP, if all dispute wheels are robust and complete, then the instance of SPP is safe.

**Lemma V.7.** *Let $S$ be an instance of the stable paths problem. If every dispute wheel of $S$ is complete and robust, then $S$ is safe.*

*Proof.* We will use proof by contradiction. Suppose every dispute wheel of $S$ is complete and robust, but $S$ is not safe.

By Lemma V.4, we know that $S$ must contain a dispute wheel that has nodes which oscillate infinitely often. Furthermore, we have assumed that this dispute wheel, $\Theta(S)$, is complete. We will show that $\Theta(S)$ can not be robust, which will be our contradiction.

Let the path assignment $\pi(0)$ and the activation sequence $\sigma$ be unsafe for $S$. We will use induction to show that we can find an initial path assignment and activation sequence such that $SPP\Theta(S)$ is not safe. As usual, let $\pi(i)$ define the path assignment at time $i$ for $S$ under the activation sequence $\sigma$. Let $t_f$ be the time where all paths have been flushed out of the system as in V.5. Let $\P(i)$ define the path assignment for SPP($\Theta(S)$) with the activation sequence $\sigma$ for all times $i \geq t_f$.

At time $t = t_f$, we let $\P(T_f)$ have the following path assignments. For all $u \in \Theta(S)$, let $\P(u, t_f) = \pi(u, t_f)$ . For all $w \notin \Theta(S)$, these nodes do not occur in SPP($\Theta(S)$).

Let $\Upsilon(i)$ be the predicate that at time $i$ the following holds true. For all $u \in \Theta(S)$, $\P(u, i) = \pi(u, i)$ .

Base Case. At time $t_f$ we let $\P(T_f)$ have the above path assignments, so we know $\Upsilon(i)$ is true.

Induction Step. Suppose $\Upsilon(i)$ is true. We know that for all $u \in \Theta(S)$, $\P(u, i) = \pi(u, i)$. Under the activation sequence $\sigma$, at time $i + 1$ the nodes $U_{i+1}$ are activated. Each node in $U_{i+1}$ will be denoted by $u_{k,i+1}$.

Suppose $u_{k,i+1} \in \Theta(S)$. This activation will cause node $u_{k,i+1}$ to take the path $\pi(u_{k,i+1}, i+1) = \text{best}(\text{choices}(\pi(i),u_{k,i+1},i))$. For node $u_{k,i+1}$, we know that $\mathcal{P}^{u_{k,i+1}}{}_{\Theta(S)} = \mathcal{P}^{u_{k,i+1}}{}_{S}$, because $\Theta(S)$ is a complete dispute wheel. Because the next hop of every availible path is in $\Theta(S)$ and by our induction hypothesis, we must then have $\text{choices}(\P(i),u_{k,i+1},i) = \text{choices}(\pi(i),u_{k,i+1},i)$ . Therefore, $\text{best}(\text{choices}(\P(i),u_{k,i+1},i)) = \text{best}(\text{choices}(\pi(i),u_{k,i+1},i))$, and $\P(u_{,i+1}, i+1) = \pi(u_{k,i+1}, i+1)$.

Suppose $u_{k,i+1} \notin \Theta(S)$. This node does not occur in $\text{SPP}(\Theta(S))$. Therefore, $\Upsilon(i) \Rightarrow \Upsilon(i+1)$.

We have used induction to show that the nodes of $\text{SPP}(\Theta(S))$ will have the same sequence of path assignments as $\pi$. We can use $\pi(t_f)$ and the subsequence of $\sigma$ beginning with the $t_f^{th}$ element with all elements $u_{k,i+1} \notin \Theta(S)$ removed as an initial path assignment and activation sequence for $\text{SPP}(\Theta(S))$ that will cause some nodes of $\Theta(S)$ to oscillate indefinitely. $\text{SPP}(\Theta(S))$ is not safe.

However, we have reached a contradiction, because we assumed every dispute wheels was robust, and thus can't be unsafe. If every dispute wheels of $S$ is complete and robust, then $S$ must be safe.

□

# E. A WEAKER SUFFICIENT CONDITION FOR SPP ROBUSTNESS

The following lemma is important for our main theorem.

**Lemma V.8.** *Let $S$ be an instance of the stable paths problem. If every dispute wheel of $S$ is robust and complete, then every dispute wheel of all subinstances of $S$ is robust and complete.*

*Proof.* Let $S = S = (G,\mathcal{P},\Lambda)$ be an an instance of the stable paths problem where $G = (V, E)$. Let $E' \subset E$ and $\text{SPP}(E')$ be a subinstance of the stable paths problem. Let $\Pi$ be any dispute wheel of $\text{SPP}(E')$. Any dispute wheel $\Pi$, must also be a dispute wheel for $S$, which can be denoted by $\Theta(S)$. Because of our assumption, $\text{SPP}(\Theta(S))$ is robust. Therefore, $\text{SPP}(\Pi)$ must also be robust, because it is a

subinstance of SPP($\Theta(S)$). Therefore $\Pi$ is a robust dispute wheel. Finally, because for each node $u \in \Pi$, $\mathcal{P}^u_{E'} \subseteq \mathcal{P}^u = \mathcal{P}^u{}_{S\ \mathrm{SPP}(\Theta(S))}$ , $\Pi$ must also be complete dispute wheel.

□

Our main theorem gives our new sufficient condition for robustness.

**Theorem V.9** (A Weaker Sufficient Condition for SPP Robustness)**.** *Let $S$ be an instance of SPP. If every dispute wheel of $S$ is complete and robust, then $S$ is robust.*

*Proof.* We know by Lemma V.8, that every dispute wheel of all subinstances of $S$ will be complete and robust. Therefore, we know by Lemma V.1 and Lemma V.7, that $S$, and all subinstances of $S$ will be uniquely solvable and safe respectively. Therefore $S$ must be robust.

□

We believe that the above results are true for SPVP, as well. We believe a similar inductive proof could be conducted for Lemma V.7 using SPVP. In such a proof, the sequence of path assignments for nodes of a complete dispute wheel would be the same as for the same nodes in the total instance of SPP.

We compare our condition for robustness to the existing sufficient condition for robustness, which is having no dispute wheel. If an instance of SPP has no dispute wheel, then it satisfies our condition. However, an instance of SPP may satisfy our condtion, but not the condition of having no disptute wheels. Therefore, our condition is weaker than the condition of having no dispute wheel. Unfortunately, our weaker sufficient condition is not a necessary and sufficient condition for robustness because there exist instances of SPP that our robust, but do not meet our condition.

Consider the instance of SPP given in Figure 22 which we will call "COUNTEREX". This instance of SPP contains the dispute wheel depicted in Figure 19 which we will call $\Pi_{\mathrm{COUNTEREX}}$. However, this dispute wheel is not complete. For COUNTEREX, the path (1 4 0) is available at node 1. However, for the derived

instance SPP($\Pi_{\text{COUNTEREX}}$), the path (1 4 0) can not be available at node 1 because node 4 does not belong to the dispute wheel. COUNTEREX does not meet our condition, but we claim COUNTEREX robust. Just because an instance of SPP is robust, this does not necessarily mean that it meets our condition. Therefore, our condition can not be necessary and sufficient.

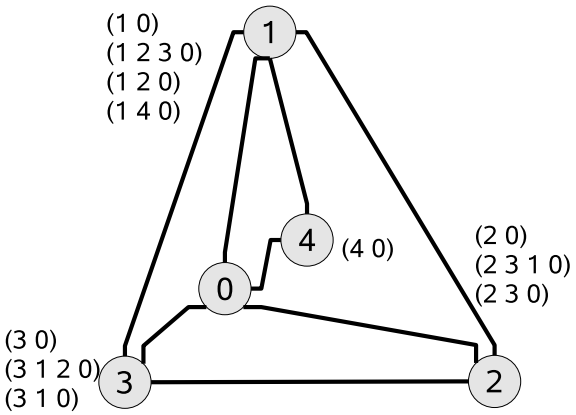


Figure 22. COUNTEREX: A Robust Instance of SPP that Does Not Meet Our Condition

Figure 23 compares our condition with the condition in previous work.

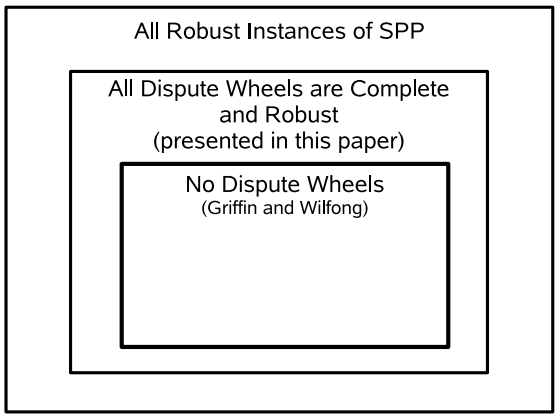


Figure 23. Conditions to Guarantee SPP Robustness

# F. APPLICATION OF MAIN THEOREM

We have shown that for any instance of SPP, if every dispute wheel is complete and robust, then the instance is robust. In order to apply this theorem, two steps must be taken. First, given an instance of SPP, we must find all dispute wheels. Second, once dispute wheels have been found, we must show that each dispute wheel is robust and complete.

## 1. Finding Dispute Wheels

In order to apply our main theorem, we must find all dispute wheels for a given instance of SPP. In Section I, we introduced Class-Based Path Vector systems as an abstraction of BGP and SPP that meet well-characterized contraints based upon the relationships between nodes. Ramachandran and Jaggard gave a centralized polynomial time algorithm (Algorithm 4.1 [Ref. 18]) that determines all directed cycles of troublesome classes which correspond to potential dispute wheels. They proved that their algorithm was complete. They left open the problem of determining exactly which dispute wheels, if any, occur for a directed cycle. We could take the troublesome cycle, and make sure that it meets some set of constraints such that if the cycle does create a dispute wheel, that dispute wheel is robust and complete. An example of such conditions are given in the next session.

Unfortunately, this approach only works for instances of the stable paths problem which meet the constraints of class-based path-vector systems.

## 2. Constraints that Guarantee Robustness Despite the Presence of a Dispute Wheel

To guarantee robustness for an instance of SPP, all dispute wheels must be robust and complete. The computational complexity of determining robustness of general instances of SPP remains an open problem [Ref. 12]. It may be NP-Hard. Therefore, we would like to develop global and local contraints that guarantee robustness, despite the presence of a dispute wheel. If all dispute wheels for an instance of SPP followed these contraints, then the instance would be guaranteed to be robust.

We introduce a set of contraints ("Set A") on SPP that is guaranteed to be robust and have a dispute wheel.

"Set A" of Contraints on $S = (G, \mathcal{P}\ , \Lambda)$

1. $V = \{d, 0, 1, ...n-1\}$ where $d$ is the origin and $n \geq 3$
2. $E = \{(1\ d), (2\ d), ...(n-1\ d), (n\ d), (1\ 2), (2\ 3), ..., (n-2\ n-1), (n\ 1)\}$
3. For each node $k \in V - \{d\}$, $\mathcal{P}^k = \{(k\ d), (k\ k+1\ d), (k\ k+1\ k+2\ d), ...(k\ k+1\ k+2...k-1\ d)\}$
4. $(k\ d)$ is the highest ranked path at every node $k$
5. For all other paths, $\lambda^k(P_1) > \lambda^k(P_2)$ if $P_1$ is longer than $P_2$.
6. For each node $k$, $k = k + n$,

If an instance of SPP meets these contraints, it will contain the dispute wheel of size $n-1$ where $u_i = i+1$, $Q_i = (i\ i+1\ 0)$, and $R_i = (i\ i+1)$. The purpose of "Set A" is to illustrate that there does exist some sets of general contraints, which guarantee robustness depsite the presence of a dispute wheel.

**Theorem V.10.** *If an instance of SPP meets "Set A" of constraints, then it is robust.*

*Proof.* We must show that the instance of SPP is uniquely solvable and safe under any combination of edge removals. If any edge $(k\ k+1)$ is removed, there is no possible way the subinstance still contains a dispute wheel, so the subinstance is safe and uniquely solvable. If all $n-1$ edges of the form $(d\ k)$ are removed, then there is a unique, safe solution where every node $k$ gets the empty path assignment. We now consider the cases where between 1 and $n-2$ edges are removed, but all edges of the form $(k\ k+1)$ are present. Without loss of generality, we assume the edge $(1\ d)$ is present. We use induction on the edges to prove that every node has a unique solution and is guaranteed to converge to it after some finite number of activations. Our induction hypothesis $\mathcal{Z}(i)$ is that node $i$ has a unique solution and is guaranteed to converge to its unique solution after some finite number of activations.

Base Case. Node 1 has the unique solution (1 $d$) because it is the highest ranked path and is also always available. Also, node 1 is activated infinitely often, so it will converge after some finite number of activations.

Inductive Step. We assume that node $i$ has a unique solution and will converge to it after some finite number of activations. We would like to show that node $i-1$ also has a unique solution and will converge to it after some finite number of activations. Suppose the edge ($i-1$ $d$) has not been removed. This case is the same as the base case, therefore $\mathcal{Z}(i-1)$ is true. Suppose the edge ($i-1$ $d$) has been removed. After some finite number of activations, node $i$ will converge to some path $P_1 = (i\ [i+1]\ [i+2]\ ...d)$. This path can not be ($i\ i+1...i-1\ d$) because edge ($i-1\ d$) is unavailable. Therefore, the path ($i-1\ i)P_1$ is available at node $i-1$ because of the "Set A" of constraints. Because this is the only available path, and the path assignment of node $i$ is unique, the path assignment of node $i-1$ must also be unique. Furthermore, because each node is activated infinitely often, node $i-1$ will be activated sometime after node $i$ recieves its path assignment, so node $i-1$ will converge after some finite number of activations as well. $\mathcal{Z}(i) \Rightarrow \mathcal{Z}(i-1)$.

By the principle of induction, all nodes in the subinstance have a unique path assignment and are guaranteed to converge to it after some finite number of activations when between 1 and $n-2$ edges of the form ($i$ $d$) fail. Therefore, if an instance of SPP meets "Set A" of contraints, it is robust under all cases of edge failures.

□

We compare how "Set A" compares with existing robust operational guidelines. Because all existing guidelines are based upon an instance of SPP having no dispute wheels, "Set A" is disjoint from existing guidelines as depicted by Figure 24. Note that the set of "All Robust Operational Guidelines" does not actually exist, because no necessary and sufficient condition for robustness has been found.

Clearly, the conditions of "Set A" create a complete dispute wheel. Any dispute wheel generated by these conditions will contain every node and the availible

paths at every node are contained in the dispute wheel

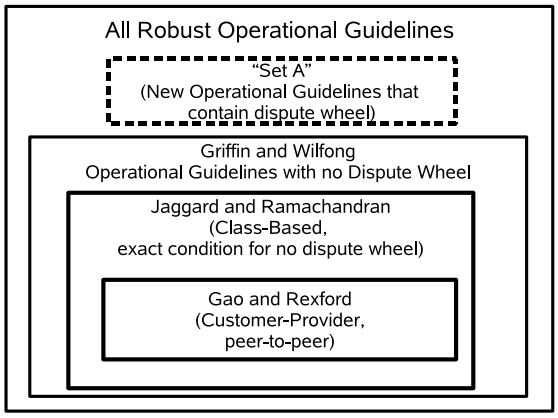


Figure 24. Robust Operational Guidelines for SPP and BGP

Unfortunately, the constraints given by "Set A," are too strict. There exist other instances of SPP that contain robust dispute wheels. We would like to investigate the most general constraints possible that guarantee the robustness and completeness of dispute wheels.

In general, our results give could be applied to give BGP operators more flexibility. Operators could follow existing operational guidelines, or they could follow new operational guidelines that are guaranteed to produce robust and complete dispute wheels. By following either such guidelines, the system of BGP routers will be provably robust.

# VI. CONCLUSION AND FUTURE WORK

In this paper, we have extended previous work on interdomain routing by focusing on the stable paths problem. In particular, we introduce a new sufficient condition for interdomain routing that guarantees robustness. This condition is weaker than those previously published. We also compare various models of BGP behavior. We show that such models do not necessarily have equivalent definitions of safety. We also show that such models do not necessarily match each other in terms of the possible path assignments each model may reach for the same instance of the stable paths problem.

There are still a large number of open problems pertaining to interdomain routing and robustness. The condition for robustness we have introduced is not likely to be the most general condition for robustness. Ramachandran conjectured that no general set of conditions can capture all robust instances of the stable paths problem (Conjecture 4.5.3 [Ref. 21]). Either a necessary and sufficient condition for the stable paths problem will have to be found, or this conjecture will need to be proven.

As mentioned in Chapter V, we believe our main results could also be proven using the simple path vector protocol. A formal proof of this would give greater confidence that our results can undoubtedly be applied to BGP.

During the research for this thesis, we were unable to prove whether or not safe (MNASM) implies safe(SPVP). Either a counterexample will need to be found, or some proof will need to be made.

The problem of devising more general conditions than those given in "Set A" remains open. These conditions are strict, and it is possible that much broader conditions based upon our main result could be given. Once broad conditions have been constructed, it would be useful to convert such conditions to guidelines for BGP operators.

THIS PAGE INTENTIONALLY LEFT BLANK

# APPENDIX. AN EXAMPLE OF A ROUTER CONFIGURATION

```
Current configuration : 1289 bytes
!
version 12.1
service timestamps debug uptime
service timestamps log uptime
no service password-encryption
!
hostname Bart
!
!
!
!
!
!
memory-size iomem 15
ip subnet-zero
!
!
!
!
interface Ethernet0/0
 ip address 10.0.6.1 255.255.255.0
!
interface Ethernet1/0
 ip address 10.0.4.1 255.255.255.0
!
interface Ethernet1/1
 ip address 10.0.1.2 255.255.255.0
!
interface Ethernet1/2
 ip address 10.64.10.1 255.255.255.0
 shutdown
!
interface Ethernet1/3
 ip address 10.2.1.17 255.255.255.248
 shutdown
!
```

```
router bgp 101
 no synchronization
 bgp log-neighbor-changes
 timers bgp 5 15
 redistribute connected
 neighbor 10.0.1.1 remote-as 102
 neighbor 10.0.1.1 route-map MARGE in
 neighbor 10.0.1.1 filter-list 103 in
neighbor 10.0.4.2 remote-as 103
 neighbor 10.0.6.2 remote-as 100
 neighbor 10.0.6.2 route-map HOMER in
 no auto-summary
!
ip classless
ip route 100.0.0.0 255.255.255.255 10.0.1.1
no ip http server
ip as-path access-list 100 permit ^100\$
ip as-path access-list 102 permit ^10
ip as-path access-list 103 deny 103
ip as-path access-list 103 permit .*
!
access-list 1 permit 10.0.1.1
route-map MARGE permit 10
 match as-path 102
 set local-preference 200
!
route-map HOMER permit 10
 match as-path 100
 set local-preference 100
!
!
!
!
line con 0
line aux 0
line vty 0 4
 login
!
end
```

# LIST OF REFERENCES


[1] D. Bertsekas and R. Gallagher. *Data Networkds, 2nd Ed.* Prentice Hall, Englewood Cliffs, NJ, 1992.

[2] Cisco. Endless bgp convergence problem in cisco ios software releases, October 2000.

[3] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms.* MIT Press, Cambridge, MA, USA, 2001.

[4] Cheng Tien Ee, Vijay Ramachandran, Byung-Gon Chun, and Scott Shenker. Resolving bgp disputes. Technical Report UCB/EECS-2006-39, EECS Department, University of California, Berkeley, April 13 2006.

[5] Nick Feamster, Ramesh Johari, and Hari Balakrishnan. Implications of Autonomy for the Expressiveness of Policy Routing. In *ACM SIGCOMM*, Philadelphia, PA, August 2005.

[6] J. Feigenbaum, R. Sami, and S. Shenker. Mechanism design for policy routing, 2003.

[7] L. Gao. On inferring autonomous system relationships in the internet, 2000.

[8] Lixin Gao and Jennifer Rexford. Stable internet routing without global coordination. In *Measurement and Modeling of Computer Systems*, pages 307–317, 2000.

[9] Mohamed G. Gouda. *Elements of Network Protocol Design.* John Wiley and Sons, Inc., New York, NY, 1998.

[10] R. Govindan, C. Alaettinoglu, G. Eddy, D. Kessens, S. Kumar, and W. Lee. An architecture for stable, analyzable internet routing. *IEEE Network Mag.*, 13:29–35, Jan./Feb. 1999.

[11] T. Griffin, A. Jaggard, and V. Ramachandran. Design principles of policy languages for path vector protocols, 2003.

[12] T. Griffin, F. Shepherd, and G. Wilfong. The stable paths problem and interdomain routing.

[13] Timothy Griffin, F. Bruce Shepherd, and Gordon T. Wilfong. Policy disputes in path-vector protocols. In *Proceedings of the 7th Annual International Conference on Network Protocols*, pages 21–30, Toronto, Canada, November 1999.

[14] Timothy Griffin and Gordon T. Wilfong. A safe path vector protocol. In *INFOCOM (2)*, pages 490–499, 2000.

[15] Timothy G. Griffin and Gordon Wilfong. On the correctness of ibgp configuration. In *SIGCOMM '02: Proceedings of the 2002 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 17–29, New York, NY, USA, 2002. ACM Press.

[16] Timothy G. Griffin and Gordon T. Wilfong. An analysis of BGP convergence properties. In *Proceedings of SIGCOMM*, pages 277–288, Cambridge, MA, August 1999.

[17] Geoff Huston. Interconnection, peering and settlements. *Internet Protocol Journal*, 2(1):2–16, March 1999.

[18] A. Jaggard and V. Ramachandran. Robustness of class-based path-vector systems, 2004.

[19] Craig Labovitz, Abha Ahuja, Abhijit Bose, and Farnam Jahanian. An experimental study of internet routing convergence technical report msr-tr-2000-08, February 2000.

[20] R. Musunuri and Cobb J.A. A complete solution for ibgp stability. In *Communications, 2004 IEEE International Conference on, Vol.2, Iss.*, pages 1177– 1181 Vol.2, june 2004.

[21] Vijay Ramachandran. *Foundations of Inter-Domain Routing*. PhD thesis, Yale University, December 2005.

[22] Y. Rekhter, T. Li, and S. Hares. A Border Gateway Protocol 4 (BGP-4). RFC 4271 (Draft Standard), January 2006.

[23] Jennifer Rexford, Jia Wang, Zhen Xiao, and Yin Zhang. Bgp routing stability of popular destinations. In *ACM SIGCOMM IMW (Internet Measurement Workshop) 2002*, 2002.

[24] Kannan Varadhan, Ramesh Govindan, and Deborah Estrin. Persistent route oscillations in inter-domain routing. *Computer Networks*, 32(1):1–16, January 2000.

[25] C. Villamizar, R. Chandra, and R. Govindan. Bgp route flap damping. RFC 2439 (Draft Standard), 1998.

# INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California

3. Professor Geoffrey Xie
   Naval Postgraduate School
   Monterey, California

4. Professor John Gibson
   Naval Postgraduate School
   Monterey, California